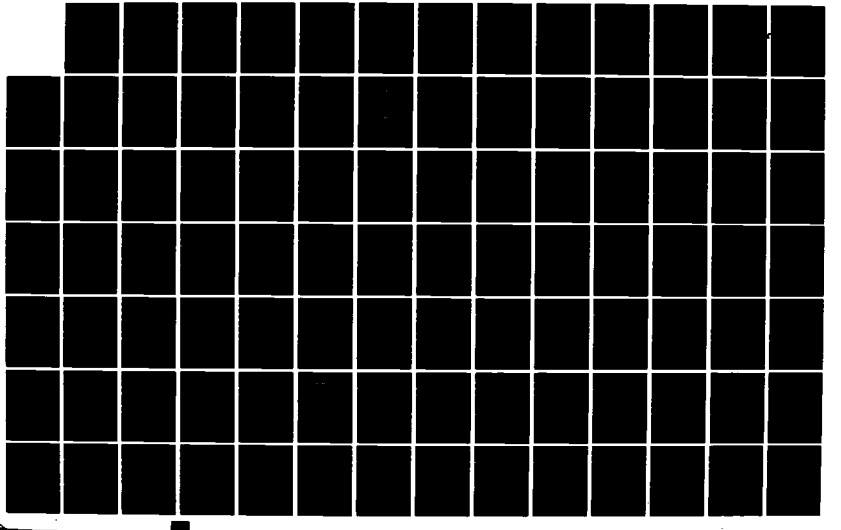
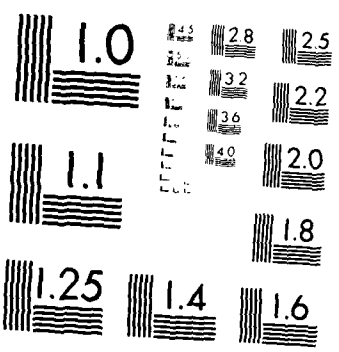
AD-A151 946 A MODERN CONTROL DESIGN METHODOLOGY WITH APPLICATION TO 1/3
THE CH-47 HELICOPTER(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH R D HOLDRIDGE JAN 85
UNCLASSIFIED AFIT/CI/NR-85-33D F/G 1/4 NL





UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-A151 946

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: AFIT/CI/NR 85-33D	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): A Modern Control Design Methodology With Application To The CH-47 Helicopter	5. TYPE OF REPORT & PERIOD COVERED: ~~THESIS~~/DISSERTATION
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Richard D. Holdridge	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: AFIT STUDENT AT: Stanford University	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: AFIT/NR, WPAFB OH 45433	12. REPORT DATE: Jan 1985
	13. NUMBER OF PAGES: 235
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASS
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

APPROVED FOR PUBLIC RELEASE: IAW AFR 190-1

LYNN E. WOLAVER
Dean for Research and
Professional Development
AFIT, Wright-Patterson AFB OH

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

ATTACHED

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE S/N 0102-LF-014-6601

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

A MODERN CONTROL DESIGN METHODOLOGY WITH APPLICATION TO THE CH-47 HELICOPTER

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS

AND THE COMMITTEE ON GRADUATE STUDIES

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

by

Richard D. Holdridge

January 1985

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

A. E. Bryson Jr.

(Principal Adviser)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

R. Cannon

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

David G. Luenberger

Approved for the University Committee on Graduate Studies:

Dean of Graduate Studies & Research

A MODERN CONTROL DESIGN METHODOLOGY WITH APPLICATION TO THE CH-47 HELICOPTER

Richard D. Holdridge, Ph.D.

Stanford University, 1985

A control system design methodology is developed which produces robust, low-order optimal controllers for multiple-input multiple-output systems. The methodology attempts to focus the strengths of recent "Modern Control" design algorithms on the problems associated with real control system designs. The methodology is a set of procedures which aids the engineer in creating a realizable controller in either digital or analog form.

To demonstrate the usefulness of the methodology, two control augmentation systems (CAS) were designed and flight tested on a CH-47 helicopter at NASA Ames Research Center. The first design was a longitudinal cruise CAS giving the pilot decoupled control of forward velocity and climb rate. This design task demonstrated the low-order controller and robustness features of the methodology. It also demonstrated the use of modern control techniques in designing integral-error controllers. Flight test results are presented. The second controller is a translational velocity command/ precision hover hold system. This two mode controller demonstrates the methodology as applied to a more complicated design task which includes control law switching and inner loop/ outer loop considerations. Flight test results are also presented.

Acknowledgements

I am grateful to my adviser, Prof. A. E. Bryson, for suggesting this research topic and then providing the enthusiastic support needed to bring it to a successful conclusion. I am equally grateful to Mr. Bill Hindson who was tireless in his efforts to make the flight test such a success. I suspect I learned as much engineering from Bill while working on the flight control system as I did in the many courses I took.

Thanks also to the many people at Ames who made the flight research possible. It was really a team effort: Dr. Vic Lebacqz, Dr. Bob Chen, Ms. Katy Hilbert who helped in the engineering and the flight test; Lt. Col. Grady Wilson and Mr. George Tucker, the project test pilots; Mr. Jim Jeske and Mr. Dave Guevara who provided the computer support. At Stanford, Doug Bernard, Peter Chu, Bruce Gardner, and Dan Rosenthal were very helpful by providing software which I incorporated into the various programs.

I'm also grateful to Prof. R. Cannon and Prof. D. Debra for reviewing this thesis and making helpful comments.

Finally, I can never adequately express the gratitude I feel to my parents, Mr. and Mrs. Kenneth B. Holdridge, whose love and support have sustained me far more than they can know.

CONTENTS

FIGURES

State Equations

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -.045 & .036 & 0 & -32.2 \\ -.37 & -2.02 & 176.0 & 0 \\ .00191 & -.0396 & -2.98 & 0 \\ 0 & 0 & 1.0 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ -28.2 & 0 \\ -11.0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_t \end{bmatrix} + \begin{bmatrix} .045 & -.036 \\ .37 & 2.02 \\ -.00191 & .0396 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_w \\ w_w \end{bmatrix} \tag{2.2}$$

Measurements

$$\begin{bmatrix} u \\ \dot{h} \\ \theta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 176.0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_t \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_w \\ w_w \end{bmatrix} + \begin{bmatrix} v_u \\ v_{\dot{h}} \\ v_\theta \end{bmatrix} \tag{2.3}$$

$$Q = \begin{bmatrix} 24.6 & 0 \\ 0 & 9.98 \end{bmatrix} \tag{2.4}$$

$$R = \begin{bmatrix} .318 & 0 & 0 \\ 0 & .318 & 0 \\ 0 & 0 & .00039 \end{bmatrix} \tag{2.5}$$

where:

- u- forward velocity, ft/sec
- w- vertical velocity, ft/sec
- q- pitch rate, $radians/sec$
- θ- pitch angle, $radians$
- δ_e- elevator, $radians$
- δ_t- throttle, ft/sec^2
- u_w- longitudinal gusts, ft/sec
- w_w- vertical gusts, ft/sec
- $v_u, v_{\dot{h}}, v_\theta$- measurement noise, $ft/sec, ft/sec, radians$
- Q- disturbance spectral density [2]
- R- measurement noise spectral density(assumes standard deviations of 1 ft/sec for u and $\dot{h}$ and 2 deg for θ with .16 second correlation times(T_c)). The spectral densities come from the approximation $S.D. \approx 2\sigma^2 T_c$.[9]

Figure 2.1: **Navion Linear Model Before Scaling.** Nominal airspeed is 100 knots (100 ft/sec). Note the large order of magnitude differences in the elements of the system matrices. This is caused by the different units of the states.

needed is a linear system in state variable form:

$$\begin{aligned} \dot{x} &= Fx + Gu + \Gamma w \\ y &= Hx + Lu + Nw + v \\ \dot{z} &= Az + By \\ u &= Cz + Dy \end{aligned} \tag{2.1}$$

where:

x- system states, $n \times 1$
z- compensator states, $r \times 1$
u- controls, $m \times 1$
w- plant disturbances, $m' \times 1$
y- sensor measurements, $p \times 1$
v- sensor noise, $p \times 1$
Q- plant disturbance spectral density matrix, $m' \times m'$
R- sensor noise spectral density matrix, $p \times p$

This model is needed for all plant conditions for which the controller is expected to operate. Typically, the conditions include the nominal operating point and several off-nominal conditions for which the control system must also be stable and perform adequately. Figure 2.1 shows the Navion model at the sea level, 104 knot flight condition. No off-nominal flight conditions are included in the example. With the dynamic model now available, the methodology can continue with the scaling of the model.

2.2. Model Scaling

Model scaling is the systematic process of changing the units of the variables in the plant model. This process is described and used by Bryson(BR,sect 7.2) and is a necessary first step in the process of compensator order reduction. Since

Chapter 2.

The Design Methodology

From the discussion of Chapter 1, the need for a structured approach to using modern control methods to design control systems is apparent. This chapter describes a design methodology developed to take advantage of the strengths of the modern control techniques while eliminating or reducing their weaknesses. The methodology is a refinement and an expansion of design procedures described by Bryson and taught in the advanced flight control course at Stanford University.[1] Figure 1.3 shows a flow chart describing the methodology. This chapter describes each step and explains why it is necessary. To facilitate the description of the methodolgy, a simple design task is traced through the process, namely the design of a longitudinal autopilot for commanding aircraft airspeed and climb rate. The plant model represents the Navion general aviation aircraft at sea level and about 100 knots(176 ft/sec) airspeed. The objective is a dynamic compensator which could be implemented on a digital compr ter in the aircraft.

2.1. Model Derivation

The development of dynamic models for use in control system synthesis is an entire engineering discipline in itself. For this design methodology, the model

dures is the lack of experience in real world application, Chapters 3 and 4 present the results of applying these techniques to a real world problem. The "real world" problem is the design of a control system for a Boeing-Vertol CH-47 "Chinook" helicopter operated by the NASA Ames Research Center. Chapter 3 shows the application of the methodology to the design of a longitudinal cruise control augmentation system(CAS). Specifically, the CAS is designed to give the pilot a velocity/climb rate command capability for cruise flight. The controller is MIMO with 4 measurements and 2 controls. Chapter 4 applies the techniques to the more difficult task of designing a position hold/velocity command controller for the CH-47 during hover and low speed operations. For this task there are two modes of operation(velocity command and position hold), switching logic for going between the modes, 11 measurements, and 3 controls which must be considered in the design. For both these designs, simulation and flight test results are presented.

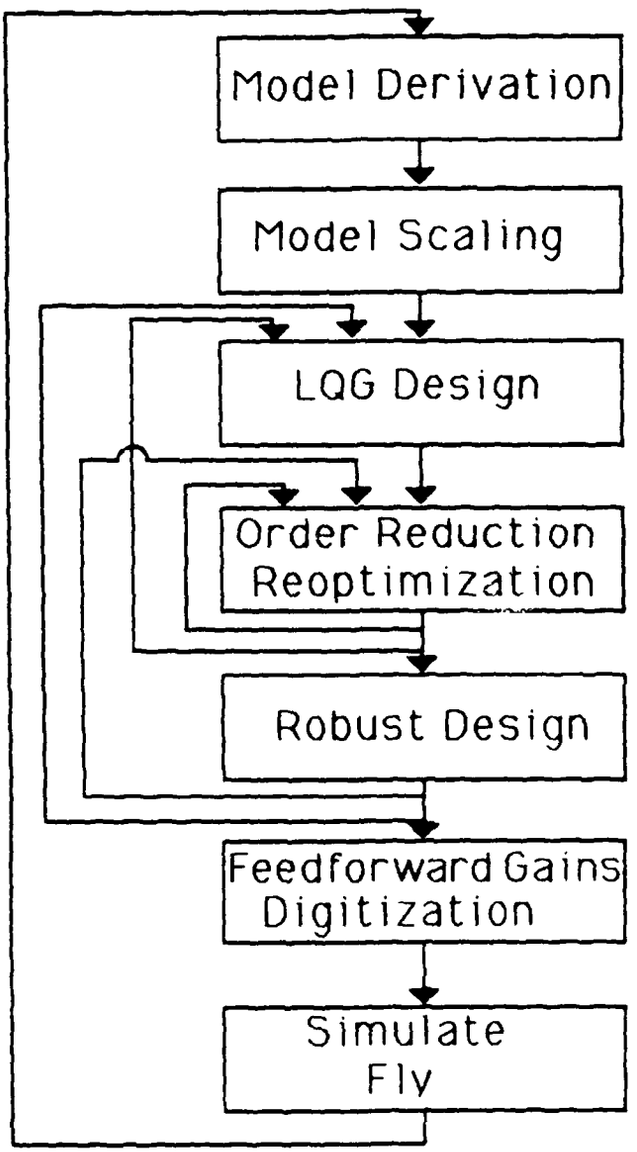


Figure 1.3: **The Design Methodology.** The many loops indicate the iterative nature of the process. The computer programs associated with each step are described in the appendices.

- The design and analysis tools for LQG techniques have only recently become easy to use in practical engineering applications.

These disadvantages provide a dilemma to the engineer faced with a difficult MIMO control design task. If he chooses to use LQG methods, he may be breaking new ground when he finally implements his system in actual operational hardware/software. On the other hand, if he uses classical techniques in his design procedure, he may end up with a design lacking in performance.

This report presents a control system design methodology which attempts to minimize the disadvantages of many modern control techniques, as described above, while taking advantage of the strengths of both modern and classical control techniques. Specifically, the methodology gives the engineer a structured approach to designing arbitrary order, robust, optimal controllers which can be implemented in digital or analog hardware. In this case, "arbitrary order" implies feedback compensation of any size, regardless of the plant model. This is an important consideration for high order plants. "Robust" is defined as insensitivity of the controller performance to changes in the plant. "Optimal" relates to the design methodology's attempt to minimize a quadratic performance index. This methodology is a refinement and expansion of techniques described by Bryson [2] and depends heavily on a robust-control design algorithm by Ly.[3] Figure 1.3 is a flow diagram showing the steps involved in the methodolgy. Chapter 2 of this report is a detailed description of this methodology. To clarify the description, a simple design is taken through all steps of the design procedures with explanations for design decisions made during the process.

Since one of the major disadvantages of using modern control design proce-

Modern (or LQG) design techniques are characterized by simultaneous closing of all loops from measurements to controls. Figure 1.2 shows the form of this type of controller. LQG design techniques offer several advantages to the designer of MIMO controllers:

- They ensure a stable design, if it exists, for the given measurements and controls.
- They result in coordinated, "graceful" controllers. In particular, the controls do not fight each other to satisfy the design requirements.
- For complicated systems, the design can be done fairly quickly.

Although these advantages make modern control design techniques very powerful, there are disadvantages which cannot be disregarded:

- The designs are often not robust to changes and uncertainties in the plant model, i.e. they are to "finely-tuned" to the plant model.
- For complex systems, the controller designs are complicated, making implementation difficult.
- Only a few designers have developed physical intuition for selecting performance indices.
- There is little physical intuition developed concerning the loops closed by the design process.
- There are relatively few examples of operational systems designed using these techniques. Practicing design engineers are therefore hesitant to use these techniques.

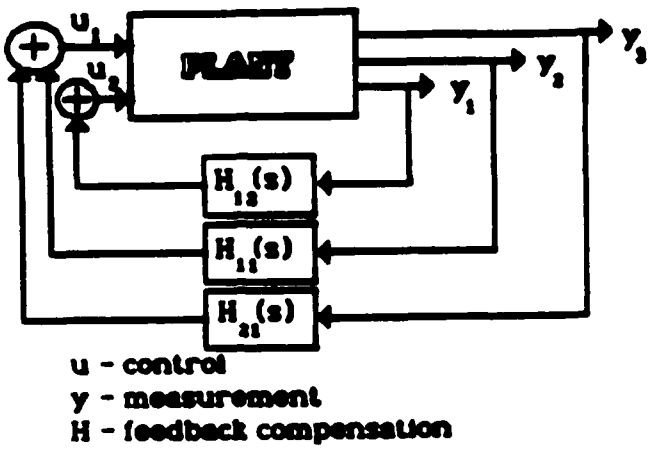


Figure 1.1: **Classical Control.** Characterized by incremental loop closures.

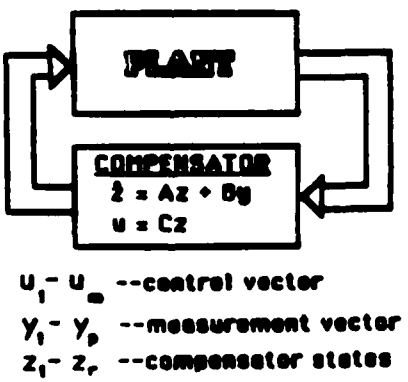


Figure 1.2: **Modern Control.** Characterized by simultaneous loop closures.

1.2. A Modern Control Design Methodology

Classical control techniques are routinely applied to the design of MIMO systems. The process is characterized by successive loop closing. (Figure 1.1) Some of the advantages of using this approach are:

- The designs are simple with clear physical intuition for each loop closed.
- For uncomplicated or open loop stable systems, the procedure is fairly fast.
- There is a large experience base for applying these methods.
- Numerous design and analysis tools exist.
- Operational hardware/software implementation is well understood and straightforward.

However, there are also some disadvantages to using classical design techniques for MIMO systems:

- For complicated or highly unstable systems, one loop may not stabilize the system.
- It may be difficult to decide what loops should be closed. This is especially true in complicated systems(large space structures for example), or in unfamiliar systems where little physical intuition has been developed.
- Different loops can "fight" each other resulting in using more control than is necessary or available.
- Many of the design techniques(Root locus, frequency response,etc.) are difficult to apply to MIMO systems.

- Better Numerical Methods(HQR, HQZ, SVD, etc.)– 1960-present

- Singular Value Techniques– Doyle and Stein and others,1970-present

- Gradient Design Methods– Ly and others,1980-present

These techniques are often labeled "modern control" where the previously mentioned methods(Root locus, Frequencly Response, etc.) are often called "classical control" techniques. Perhaps the major difference between the two is that classical techniques are best suited to single input single output(SISO) dynamic systems where the modern control techniques are equally applicable to multiple input multiple output(MIMO) systems.

Virtually all control systems in operational use today, in numerous diverse applications, have been designed using classical control techniques. These techniques have been preferred over the modern control methods because most of the applications are fairly simple SISO systems. Even if the systems being controlled aren't truly SISO, acceptable SISO approximations can often be made. For instance, aircraft motions can be separated into longitudinal and lateral motions which can be approximated by fast and slow linear SISO systems. Design methodologies based on classical control use these SISO approximations for autopilot designs. As we might expect, this approach results in less performance than might be possible had we not made the simplifying assumptions (necessary to use the design techniques). This procedure is completely inappropriate if the system to be controlled cannot be adequately approximated by several uncoupled SISO systems.

engineers had the opportunity to build greatly improved control systems. However, the existing analytical tools were difficult to apply. More systematic analysis and design tools were needed to aid in developing control systems. Such techniques were developed in the early part of this century. A few of these techniques(applicable to linear systems) are listed below:

- Nyquist Stability Analysis– Nyquist, 1930's
- Frequency Response– Bode, 1940's
- Root Locus– Evans, 1940's

McRuer's Chapter 1 contains an excellent summary of these early analytical techniques and their applications.[1] All these techniques(plus others) provided the theory, and the signal processing devices existed to design sophisticated control systems which were installed in operational systems such as aircraft, missiles, rockets, ships, etc. The requirements of the military during World War II pushed the development and refinement of these techniques and of hardware capable of implementing the designs.

During the 1950's, the tools and hardware available to control engineers continued to expand. It was during this decade that the digital computer became a practical tool for designing and implementing control systems. With the digital computer, the control design engineer had much more signal processing capability. Analysis and design techniques soon developed which took advantage of the digital computer. A few of these techniques are:

- Kalman Filter– Kalman,1960
- Linear Quadratic Gaussian(LQG) techniques– Bryson and others,1960-present

Chapter 1.

Introduction

1.1. Historical Background on Control System Design

Control system design, as an engineering discipline, is relatively new. It has been coupled closely with the development of measurement devices(sensors), electronic signal processing capability(analog and digital hardware), and the development of mathematical techniques for the design and analysis of dynamic systems. Before electronic signal processing, ingenious mechanical devices were designed which incorporated simple feedback mechanisms. "Simple" here implies their analytical complexity; sometimes these devices could be quite compicated mechanically. Some simple examples of these mechanical feedback devices are:

- Water level control in a tank using float valves
- James Watts' centrifugal governor for steam engines

Conventional techniques for analyzing dynamic systems (Newtonian dynamics, Lagrangian dynamics, etc.) were adequate to study and design these mechanical devices.

With the development of electronic signal processing and electronic sensors,

many dynamic systems, including the example here, have parameters with different units, it is often difficult to compare control gains or estimator gains for states or measurements with differing units. In the Navion example, to determine the relative importance of velocity to elevator and pitch angle to elevator gains would be difficult since velocity has units of ft/sec and pitch angle has units of radians. Model scaling allows the engineer to select a new set of units which result in the plant model having variables with units having a sort of "equivalent" importance to the designer. Of course, selection of such a set of units requires an understanding of the dynamics of the process being controlled. For the Navion, the angle variables are changed from *radians* to .01 *radians* while the linear variables remain in units of ft/sec. Such a selection is reasonable since .01 *radians* and 1 ft/sec are similar in importance to a pilot flying the aircraft. For a supersonic aircraft, a suitable selection might be .01 *radians* and 10 ft/sec.

Figure 2.2 shows the scaled dynamic model. This change of scaling is accomplished analytically using similarity transformations based on the changes in units. Appendix A presents the derivation of the transformations and the equations used to transform all matrices of the dynamic model into the new units. The computer program ROPTSYS, described in Appendix J, implements these equations as the first step in calculating an optimal full order compensator. The next step in the design methodology uses this scaled dynamic system as a starting point.

2.3. LQG Design

During this step in the design procedure, the optimal regulator and estimator gains are calculated for the scaled dynamic system. Appendix B shows the deriva-

State Equations

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \end{bmatrix}_{scaled} = \begin{bmatrix} -.045 & .036 & 0 & -.322 \\ -.37 & -2.02 & 1.76 & 0 \\ .191 & -3.96 & -2.98 & 0 \\ 0 & 0 & 1.0 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix}_{scaled} + \begin{bmatrix} 0 & 1 \\ -.282 & 0 \\ -11.0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_t \end{bmatrix}_{scaled} + \begin{bmatrix} .045 & -.036 \\ .37 & 2.02 \\ -.191 & 3.96 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_w \\ w_w \end{bmatrix}_{scaled} \tag{2.6}$$

Measurements

$$\begin{bmatrix} u \\ \dot{h} \\ \theta \end{bmatrix}_{scaled} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1.76 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix}_{scaled} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_t \end{bmatrix}_{scaled} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_w \\ w_w \end{bmatrix}_{scaled} + \begin{bmatrix} v_u \\ v_{\dot{h}} \\ v_\theta \end{bmatrix}_{scaled} \tag{2.7}$$

$$Q_{scaled} = \begin{bmatrix} 24.6 & 0 \\ 0 & 9.98 \end{bmatrix}$$

$$R_{scaled} = \begin{bmatrix} .318 & 0 & 0 \\ 0 & .318 & 0 \\ 0 & 0 & 3.9 \end{bmatrix} \tag{2.8}$$

where:

- u- forward velocity, ft/sec
- w- vertical velocity, ft/sec
- q- pitch rate, .01 *radians/sec*
- θ- pitch angle, .01 *radians*
- δ_e- elevator, .01 *radians*
- δ_t- throttle, ft/sec^2
- u_w- longitudinal gusts, ft/sec
- w_w- vertical gusts, ft/sec
- $v_u, v_{\dot{h}}, v_\theta$- measurement noise, $ft/sec, ft/sec, .01$ *radians*
- Q- disturbance spectral density, (From Bryson [2] sect 9.4)
- R- measurement noise spectral density(assumes standard deviations of 1 ft/sec for u and $\dot{h}$ and 2 deg for θ with .16 second correlation times(T_c). The spectral densities come from the approximation, $S.D. \approx 2\sigma^2 T_c$.[9]

Figure 2.2: **Navion Linear Model After Scaling.** The results of the scaling are most noticeable in the F, G, and R matrices.

tion of the technique used to calculate these gains. The optimal gain calculation of this appendix is based on the eigenvector decomposition of the Hamiltonian matrix as described in Hall and Bryson.[17] A modification to the technique is the capability to weight the linear combination of states and controls, $y_o = H_o x + Du$, in determining regulator gains, and, by duality, to include plant noise in the measurement, $y_m = H_m x + Lu + Nw + v$, in determining estimator gains.[1] This modification is essential for aerospace applications since it allows acceleration to be used as a measurement for estimator design, and weighted in regulator design. The ROPTSYS computer program, described in Appendix J, calculates these regulator and estimator gains. It also calculates the compensator dynamic system based on these gains. One item of note is that the A and B matrices, used in the quadratic performance index($P.I. = \int_0^\infty (y_o^T A y_o + u^T B u)\, dt$), can start as the identity matrices since the system has been scaled in "equivalent" engineering units.

The compensator dynamic system is displayed in three formats:

Conventional Form

$$\begin{aligned} \dot{z} &= (F - GC - KH)z + Ky \\ u &= Cz \end{aligned} \tag{2.9}$$

where:

z- compensator states, $n \times 1$

y- sensor measurements, $p \times 1$

u- controls, $m \times 1$

K- steady state Kalman filter gains, $n \times p$

C- optimal regualtor gains, $m \times n$

[1] Franklin and Powell show this derivation for the direct digital design case.[6]

Modal Form

$$\begin{aligned} \dot{z} &= F_{mod}\, z + K_{mod}\, y \\ u &= C_{mod}\, z \end{aligned} \tag{2.10}$$

where:

F_{mod}- modal form of $(F - GC - KH)$

K_{mod}- modal form of K

C_{mod}- modal form of C

$$F_{mod} = \begin{bmatrix} \sigma_1 & \omega_1 & 0 & \cdots \\ -\omega_1 & \sigma_1 & & \\ 0 & & \sigma_2 & \omega_2 & \cdots \\ & & -\omega_2 & \sigma_2 & \\ \vdots & & \vdots & & \ddots \end{bmatrix}, n \times n$$

$$K_{mod} = \begin{bmatrix} \star & \star & \cdots \\ \star & \star & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}, n \times p \tag{2.11}$$

$$C_{mod} = \begin{bmatrix} \star & \star & \cdots \\ \star & \star & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}, m \times n$$

Block Minimal Form

$$\begin{aligned} \dot{z} &= F_{min}\, z + K_{min}\, y \\ u &= C_{min}\, z \end{aligned} \tag{2.12}$$

where:

F_{min}- minimal form of $(F - GC - KH)$

K_{min}- minimal form of K

C_{min}- minimal form of C

$$
\begin{aligned}
F_{min} &= \begin{bmatrix} 0 & 1 & & 0 & \dots \\ a_1 & a_2 & & & \\ & 0 & 0 & 1 & \dots \\ & & a_1 & a_2 & \\ \vdots & & \vdots & & \ddots \end{bmatrix}, n \times n \\
K_{min} &= \begin{bmatrix} \star & \star & \dots \\ \star & \star & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}, n \times p \\
C_{min} &= \begin{bmatrix} \star & \star & 0 & 1 & \dots \\ 0 & 1 & \star & \star & \dots \\ \star & \star & \star & \star & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, m \times n
\end{aligned}
\tag{2.13}
$$

The modal form is calculated using the eigenvector matrix as a similarity transformation. Appendix C shows the equations needed to do this. The transformation to minimal form is derived in Appendix C. The modal and minimal realizations are important since they are unique realizations of the dynamic system. From the block minimal realization of the compensator, the designer can assess the relative importance of the different modes. To aid in this determination of modal importance, the ROPTSYS program also calculates two input/output measures, both based on work done by Bernard.[7] One measure is the singular value of the residue matrix associated with that mode. It is defined as:

$$M_1 = \overline{\sigma}(R_i) = \sqrt{g_1 g_1^T + g_2 g_2^T}\sqrt{h_1^T h_1 + h_2^T h_2} \tag{2.14}$$

where:

g_1, g_2– the two rows of K_{min} associated with the mode being analyzed
h_1, h_2– the two columns of C_{min} associated with the mode being analyzed
$\overline{\sigma}(R_i)$– the singular value of the residue matrix for the i^{th} mode

The second measure of merit, M_2, is just the first weighted by the real part of the mode being evaluated:

$$M_2 = \frac{\overline{\sigma}(R_i)}{|\sigma_i|} \tag{2.15}$$

The larger these measures are, the more important that mode is to the overall input/output characteristics of the compensator. In the case of unstable compensator modes, these should not be eliminated since they are usually required for stabilization of unstable plant modes. Figure 2.3 shows the results for the Navion control, which came from ROPTSYS. The input data required to get these results are shown in Appendix J.

From these results, we note that the open loop Navion has a well damped short period (the fast open loop mode) but the phugoid (the slow open loop mode) is only lightly damped. The regulator design improves these dynamics by forcing the two phugoid roots onto the real axis and speeding them up. The resulting compensator is fairly fast (as fast as the aircraft response itself) and well damped. This design was also analyzed using the RSANDY program, described in Appendix K, to evaluate the compensator performance in the presence of plant disturbances (vertical and longitudinal wind gusts) and measurement noise. The wind gust data come from a turbulence model described by Bryson.[1] The performance is tabulated in Figure 2.4.

Using these data, Figures 2.3 and 2.4, we can proceed to the next step of the design, simplification of the full-order compensator.

2.4. Compensator Order Reduction and Reoptimization

As with each step in this methodology, the subject of compensator order reduction and simplification is an engineering discipline in itself. For this methodology, the compensator mode measures, M_1 and M_2, are the criteria for deciding which

System	Eigenvalues	Damping
Open Loop(F)	$-2.51 \pm j2.59, -.017 \pm j.21$	.7, .08
Regulator(F-GC)	$-2.43 \pm j3.05, -1.03, -2.43$	.62, 1, 1
Estimator(F-KH)	$-11.3, -2.98, -.84 \pm j.43$	1, 1, .89
Compensator(F-GC-KH)	$-11.3, -3.04 \pm j2.34, -1.82$	1, .8, 1

Mode	Real	Imag	M_1	M_2
1	-11.34	0	2.62	.23
2	-3.04	2.34	1.86	.61
3	-3.04	-2.34	–	–
4	-1.82	0	.83	.46

Compensator Dynamic System Matrices

$$F_{min} = \begin{bmatrix} -11.342 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -14.686 & -6.0721 & 0 \\ 0 & 0 & 0 & -1.8155 \end{bmatrix}$$

$$K_{min}(scaled) = \begin{bmatrix} -.0133 & -2.6218 & .0244 \\ -.278 & -.0886 & .0652 \\ 1.49 & .872 & -.337 \\ .771 & -.0464 & -.101 \end{bmatrix}$$

$$K_{min}(unscaled) = \begin{bmatrix} -.0133 & -2.6218 & 2.44 \\ -.278 & -.0886 & 6.52 \\ 1.49 & .872 & -33.7 \\ .771 & -.0464 & -10.1 \end{bmatrix} \qquad (2.16)$$

$$C_{min}(scaled) = \begin{bmatrix} 1.0 & 0 & 1.0 & -.36 \\ -.0344 & .198 & .0216 & 1.0 \end{bmatrix}$$

$$C_{min}(unscaled) = \begin{bmatrix} -.01 & 0 & -.01 & .0036 \\ .0344 & -.198 & -.0216 & -1.0 \end{bmatrix}$$

Figure 2.3: **Navion Full Order Compensator.** The unscaled matrices are in units of the physical system.

Standard Deviations						
Controls		States				
δ_e (deg)	δ_t (deg/sec^2)	u (ft/sec)	w (ft/sec)	q (deg/sec)	θ (deg)	h (ft/sec)
.153	.284	.5	2.25	.66	.47	1.32

Figure 2.4: **Navion Full Order Compensator Performance.** This statistical performance is based on the noise characteristics described in Figure 2.1.

modes of a compensator can be eliminated. The M_2 term seems to be most useful in the work here on flight vehicles. A mode can probably be eliminated with little performance impact if its value of $M_2/M_{2largest}$ is .1 or less. When this ratio is greater than .1, that mode may still be neglectable but the decision must rest on reduced order analysis results. In addition to eliminating modes, a compensator can also be simplified by eliminating unimportant measurements. This is desirable since it would lower the cost of the control system. The scaled K_{min} matrix provides the data which are used to decide if a specific measurement is necessary. If the magnitudes of the elements of a specific column of K_{min} are smaller than the other column elements, the measurement associated with that column may be neglectable. Of course, the final decision must be based on the simplified compensator analysis results.

Applying these design guidelines to the Navion example, Figure 2.3, we note the two real modes have an M_2 smaller than the single complex mode so we eliminate these modes and the associated rows of K_{min} and columns of C_{min}. Examining the values of scaled K_{min}, we note the column associated with the θ (pitch angle) measurement is smaller than the other so we try eliminating this measurement. The importance of the scaling process is evident here since the θ column elements in the unscaled K_{min} matrix have large magnitudes due to the radian units. Thus, without scaling, the θ measurement would have seemed more important than it was. The compensator is now second order, 2 input, and 2 output, a significant simplification from the fourth order, 3 input, and 2 output full order compensator.

The next step is to analyze and reoptimize the reduced order compensator

Simplified Compensator before Reoptimisation

$$\begin{bmatrix}\dot{z}_1\\ \dot{z}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ -14.69 & -6.072\end{bmatrix}\begin{bmatrix}z_1\\ z_2\end{bmatrix} + \begin{bmatrix}-.278 & -.0886\\ 1.485 & .873\end{bmatrix}\begin{bmatrix}u\\ \dot{h}\end{bmatrix}$$
$$\begin{bmatrix}\delta_e\\ \delta_t\end{bmatrix} = \begin{bmatrix}0 & -.01\\ -.1978 & -.0216\end{bmatrix}\begin{bmatrix}z_1\\ z_2\end{bmatrix} \qquad (2.17)$$

Simplified Compensator after Reoptimisation

$$\begin{bmatrix}\dot{z}_1\\ \dot{z}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ -13.59 & -16.93\end{bmatrix}\begin{bmatrix}z_1\\ z_2\end{bmatrix} + \begin{bmatrix}-.1366 & .2124\\ 2.676 & -3.241\end{bmatrix}\begin{bmatrix}u\\ \dot{h}\end{bmatrix}$$
$$\begin{bmatrix}\delta_e\\ \delta_t\end{bmatrix} = \begin{bmatrix}0 & -.01\\ -13.14 & -1.228\end{bmatrix}\begin{bmatrix}z_1\\ z_2\end{bmatrix} \qquad (2.18)$$

Full Order vs Reduced Order Performance

System	Standard Deviation						
	δ_e (deg)	δ_t (deg/sec^2)	u (ft/sec)	w (ft/sec)	q (deg/sec)	θ (deg)	$\dot{h}$ (ft/sec)
Full Order	.153	.284	.5	2.25	.66	.47	1.32
Reduced Order	.2	.31	.57	2.39	.77	.58	1.23

Figure 2.5: **Navion Reduced Order Compensator Design Results.** The reduced order compensator was unstable before the reoptimization. The performance is based on the disturbance and noise properties of Figure 2.1

using the RSANDY computer program. The RSANDY program, described in Appendix K, is a modified version of a design and analysis code written by Ly.[8] The program designs robust, low order compensators using a gradient search technique based on a quadratic performance index. The results of using this code on the Navion simplified compensator are shown in Figure 2.5.

The reduced order compensator resulted in an unstable closed loop system. This points to the necessity of the reoptimization step after compensator simplification. The utility of Ly's code is also evident here since it allows unstable initial guesses when reoptimizing the compensator. Another important feature,

not used in this example, is the capability of including several plant conditions in the optimization to ensure compensator stability robustness to changes in plant parameters. The selection of weighting matrices in the quadratic performance index, used by the RSANDY program, was based on "Bryson's Rule" as applied to the scaling parameters. Briefly, "Bryson's Rule" suggests that the outputs and controls be weighted by the inverse square of their scale factors. For this example, the outputs were u(velocity) and $\dot{h}$(climb rate), and the controls were δ_e(elevator) and δ_t(throttle), with units of ft/sec, ft/sec, $radians$, and ft/sec^2, respectively. The scaled units were ft/sec, ft/sec, .01 $radians$, and ft/sec^2. The performance index is then:

$$P.I. = \int_0^{t_f} (y^T Q y + u^T R u) dt \tag{2.19}$$

where:

$$\begin{aligned} y &= \begin{bmatrix} u \\ \dot{h} \end{bmatrix} \\ u &= \begin{bmatrix} \delta_e \\ \delta_t \end{bmatrix} \\ Q &= \begin{bmatrix} \frac{1}{1^2} & 0 \\ 0 & \frac{1}{1^2} \end{bmatrix} or \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ R &= \begin{bmatrix} \frac{1}{(.01)^2} & 0 \\ 0 & \frac{1}{1^2} \end{bmatrix} or \begin{bmatrix} 10000 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned} \tag{2.20}$$

For more complicated problems, these weighting terms may need to be adjusted to get the desired performance. The input file for RSANDY used to get the results of Figure 2.5 is listed in Appendix K.

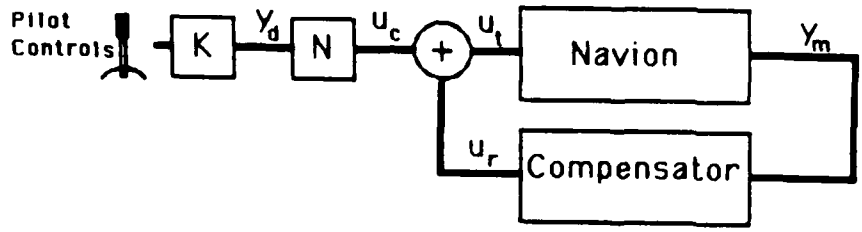
The results of Figure 2.5 show what might be expected. The simplified compensator has slightly degraded performance in that both the aircraft motion and control activity is slightly larger than the full order compensator. These statistical

performance parameters are important since they indicate the disturbance rejection properties of the design but we recall the design objective was a velocity and climb rate *command* system for the Navion. Since the designs thus far are only regulators, we still need to design the command capability for the controller.

2.5. Command Inputs

Generally, control systems are expected to do more than regulate the outputs of a process to some nominal value. We need the ability to change the output of the compensated system to any selected realizable value. To accomplish this end, the desired outputs are used to determine actuator commands which are the steady state controls for the desired outputs. Figure 2.6 shows this concept. The maximum number of outputs which can be commanded is the same as the number of independent controls in the system. This means that the feedforward matrix is square. An additional constraint is that the desired outputs must be physically realizable. For instance, velocity and position cannot be decoupled, i.e. you cannot command a steady velocity while holding position. Appendix D derives the algorithm which calculates this decoupling feedforward matrix for the case of equal numbers of controls and desired outputs. This algorithm is implemented in the SETPNT computer program described in Appendix L.

To complete the Navion autopilot design, we need to include the climb rate and forward velocity command capabilities required by the specifications. For this aircraft, these two outputs can be decoupled (controlled independently) with the two controls, elevator and throttle. Using the SETPNT program with the Navion data(shown in Appendix L), the feedforward matrix for the full order compensator



Y_d -- Desired Output = [velocity, climb rate]

Y_m -- Measurements = [velocity, climb rate]

u_c -- Command Control = [elevator, throttle]

u_r -- Regulator Control = [elevator, throttle]

u_t -- Total Control = [elevator, throttle]

K -- Control Sensitivities

N -- Feedforward Gains

Figure 2.6: **Commanding a Desired Output.** The K matrix is diagonal and based on human factor considerations. The N matrix comes from an analysis of the system in steady state; the derivation is described in Appendix D.

is:

$$N = \begin{bmatrix} .002165 & -.001578 \\ .4505 & .1398 \end{bmatrix} \tag{2.21}$$

For the reduced order controller, the result is:

$$N = \begin{bmatrix} .002243 & -.002124 \\ .5354 & .2653 \end{bmatrix} \tag{2.22}$$

where:

$$\begin{bmatrix} \delta_e(radians) \\ \delta_t(ft/sec^2) \end{bmatrix} = N \begin{bmatrix} u_{desired}(ft/sec) \\ \dot{h}_{desired}(ft/sec) \end{bmatrix} \tag{2.23}$$

One item to note here is the fact that the feedforward matrices are dependent on both the plant and the controller. The correctness of these decoupling feedforward matrices becomes clear when the Navion designs are evaluated in simulation.

2.6. Simulation and Test

The final step of any design methodology is the demonstration of performance. The first step in such a demonstration is the use of a simple linear simulation. Once the design is validated in linear simulation, the simulation might be expanded to model some of the important nonlinearities in the physical system. In this methodology, the RSANDY program, described in Appendix K, includes the option of creating a linear simulation model. This model is then used by the SIMPLOT program (described in Appendix M) to simulate the closed or open loop system with or without the random disturbances. Using these RSANDY and SIMPLOT programs the Navion full and reduced order compensators can be compared.

The results of step commands in velocity and climb rate are shown in the following figures. Comparing Figures 2.7 and 2.8 we note the performance is

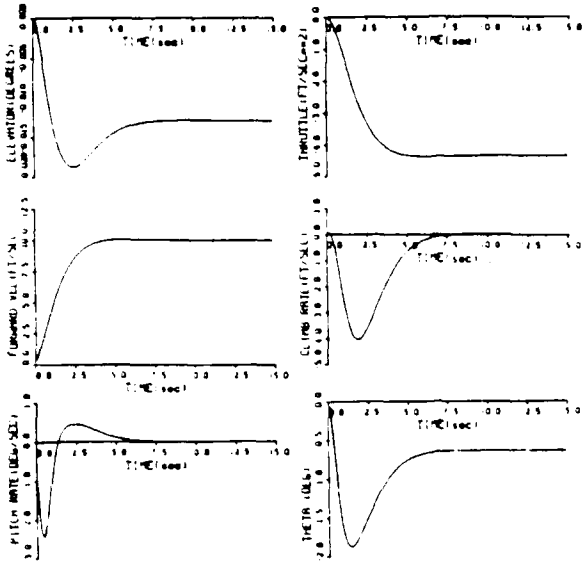


Figure 2.7: **Velocity Response for Full Order Compensator.** The system has a rise time of about 2.5 seconds and near critical damping.

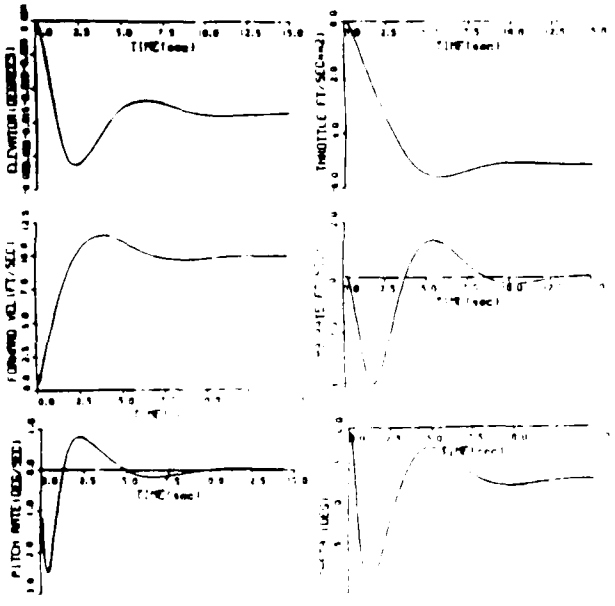


Figure 2.8: **Velocity Response for Reduced Order Compensator.** The rise time is about 2.5 seconds but the damping is only about .5.

unscaled model q and θ appear much less noisy than the u and $\dot{h}$ measurements indicating the possibility that u and $\dot{h}$ may not even be necessary. After scaling, the spectral densities are closer in magnitude but θ is still more accurate. With the scaling accomplished, the methodology continued with the design of the full order compensator using the ROPTSYS program.

3.4. LQG Design of the Full Order Compensator

The design of the full order compensator is straightforward since we have all the data necessary to run the design program ROPTSYS. Since we have the system scaled, the first choices for output and control weighting matrices in the regulator design are identity matrices. With this selection, the compensator is calculated and shown in Figure 3.6. Examining the closed loop roots (the estimator and regulator roots) we note the very slow estimator pole at -.0088. The slow estimator pole results from having a very accurate measurement of θ (low noise spectral density) but relatively noisy measurements of u and $\dot{h}$. The fast filter roots estimate mostly θ and q while the slow mode estimates mainly u. If we had introduced $\dot{h}$ and u commands into the compensator(Equation 3.9), the slow estimator mode would not have affected the response to commands, since the estimator would have been "tracking" closely before the commands. For perfect tracking, the estimator modes are not excited at all. This need to include the command in the compensator was not recognized until later so the responses of Figure 3.7 show the poor performance when commands are *not* properly fed forward to the compensator. Figure 3.8 shows

State Equations

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \\ \dot{\delta}_e \\ \dot{\delta}_c \end{bmatrix} = \begin{bmatrix} -.0204 & .0377 & .0236 & -.3217 & .0127 & .0426 \\ -.0663 & -.551 & .9902 & -.0186 & .0467 & -.936 \\ -.42 & 1.76 & -1.682 & 0 & 3.91 & 1.53 \\ 0 & 0 & 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -40.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -40.0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \\ \delta_e \\ \delta_c \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 40.0 & 0 \\ 0 & 40.0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_c \end{bmatrix} + \begin{bmatrix} .0205 & -.037 \\ .0663 & .5512 \\ .42 & -1.76 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_w \\ w_w \end{bmatrix} \tag{3.6}$$

Measurements

$$\begin{bmatrix} q \\ \theta \\ \dot{h} \\ u \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 100.0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_c \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_w \\ w_w \end{bmatrix} + \begin{bmatrix} v_q \\ v_\theta \\ v_{\dot{h}} \\ v_u \end{bmatrix} \tag{3.7}$$

$$Q = \begin{bmatrix} 24.6 & 0 \\ 0 & 9.98 \end{bmatrix}$$

$$R = \begin{bmatrix} .323 & 0 & 0 & 0 \\ 0 & 8.1 \times 10^{-4} & 0 & 0 \\ 0 & 0 & 16.0 & 0 \\ 0 & 0 & 0 & 1.1 \end{bmatrix} \tag{3.8}$$

where:

u	forward velocity, ft/sec
w	vertical velocity, ft/sec
q	pitch rate, .01 $radians/sec$
θ	pitch angle, .01 $radians$
δ_e	pitch control, .1 $inches$
δ_c	collective lever, .1 $inches$
u_w	longitudinal gusts, ft/sec
w_w	vertical gusts, ft/sec
$v_q, v_\theta, v_{\dot{h}}, v_u$	measurement noise, .01 $radians/sec$, .01 $radians$, ft/sec, ft/sec
Q	disturbance spectral density, [1]
R	measurement noise spectral density(assumes standard devations shown in Figure 3.1 with 3 Hz bandwidth) The spectral densities come from the approximation $S.D. \approx 2\sigma^2 T_c$. [9]

Figure 3.5: **CH-47 Scaled Longitudinal Linear Model at 60 knots.** Even after scaling, the θ measurement is much more accurate than the others.

State Equations

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \\ \dot{\delta}_e \\ \dot{\delta}_c \end{bmatrix} = \begin{bmatrix} -.0204 & .0377 & 2.36 & -32.17 & .127 & .426 \\ -.0663 & -.551 & 99.02 & -1.86 & .467 & -9.36 \\ -.0042 & .0176 & -1.682 & 0 & .391 & .153 \\ 0 & 0 & 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -40.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -40.0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \\ \delta_e \\ \delta_c \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 40.0 & 0 \\ 0 & 40.0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_c \end{bmatrix} + \begin{bmatrix} .0205 & -.037 \\ .0663 & .5512 \\ .42 & -1.76 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_w \\ w_w \end{bmatrix} \tag{3.3}$$

Measurements

$$\begin{bmatrix} q \\ \theta \\ \dot{h} \\ u \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 100.0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \\ \delta_e \\ \delta_c \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_c \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_w \\ w_w \end{bmatrix} + \begin{bmatrix} v_q \\ v_\theta \\ v_{\dot{h}} \\ v_u \end{bmatrix} \tag{3.4}$$

$$Q = \begin{bmatrix} 24.6 & 0 \\ 0 & 9.98 \end{bmatrix}$$

$$R = \begin{bmatrix} 3.23 \times 10^{-5} & 0 & 0 & 0 \\ 0 & 8.1 \times 10^{-8} & 0 & 0 \\ 0 & 0 & 16.0 & 0 \\ 0 & 0 & 0 & 1.1 \end{bmatrix} \tag{3.5}$$

where:

u	forward velocity, *ft/sec*
w	vertical velocity, *ft/sec*
q	pitch rate, *radians/sec*
θ	pitch angle, *radians*
δ_e	pitch control, *inches*
δ_c	collective lever, *inches*
u_w	longitudinal gusts, *ft/sec*
w_w	vertical gusts, *ft/sec*
$v_q, v_\theta, v_{\dot{h}}, v_u$	measurement noise, *radians/sec, radians, ft/sec, ft/sec*
Q	disturbance spectral density, [1]
R	measurement noise spectral density(assumes standard devations shown in Figure 3.1 with 3 Hz bandwidth) The spectral densities come from the approximation $S.D. \approx 2\sigma^2 T_c$. [9]

Figure 3.4: **CH-47 Longitudinal Linear Model at 60 knots.** The highly accurate θ measurement comes from the inertial navigation system(INS), the pitch rate from a body mounted rate gyro, the velocity from the pitot-static system, and the climb rate from an instantaneous vertical speed indicator (IVSI).

other consideration in the development of a linear model for this design is the need to avoid "nuisance disengages" of the experimental control system in the CH-47. These automatic disengages are a safety feature of the modified control system which cause the experimental control system to be tripped off when the control rates exceed a known rate. The approach used to eliminate the possibly of these problems was to include simple first order actuator models:

$$\dot{u}_{actual} = -40\, u_{actual} + 40\, u_{desired} \tag{3.2}$$

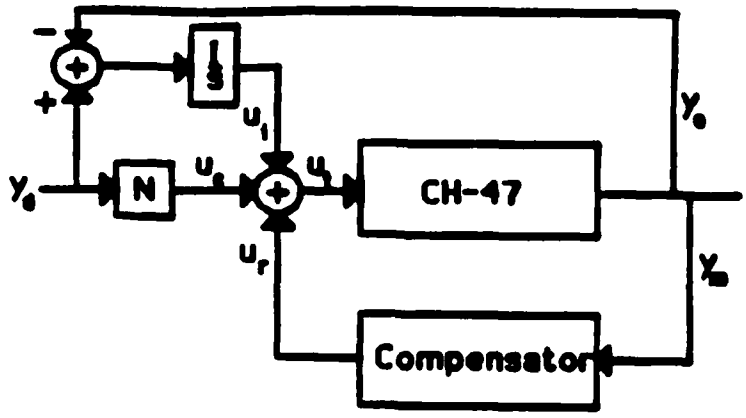
By weighting $\dot{u}_{actual}$ in the performance index, the control rates should not get large, hence eliminating the nuisance disengages.[1] Before starting the control design, the linear model was scaled to help in compensator order reduction.

3.3. Model Scaling

The selection of units for the CH-47 at a 60 knot flight condition is dependent on our understanding of the aircraft dynamics. In this case, 1 ft/sec of velocity or climb rate is about as important to the pilot as .01 *radian* of pitch angle or .01 $radian/sec$ of pitch rate. Similarly, the pilot would feel comfortable using .1 *inch* of either control to command the 1 ft/sec of velocity. Another way of looking at this scaling is to note that a .01 *radian* pitch angle would result in a 1 ft/sec climb rate at the nominal airspeed of 60 knots($\approx 100 ft/sec$). Figures 3.4 and 3.5 show the linear model at 60 knots both before and after scaling the system into the units above.

The usefulness of the scaling step is again evident if we look at the measurement noise spectral density matrix for the scaled and unscaled models. In the

[1] Flight test later confirmed that the experimental system seldom had these type disengages.



Y_a -- Actual Outputs (Subset of Measurements)

Y_d -- Desired Outputs

Y_m -- Measurements

u_c -- Command Control

u_i -- Integral Control

u_r -- Regulator Control

u_t -- Total Control

Figure 3.3: **Controller Structure for The Longitudinal Cruise Autopilot.** Flexibility comes from making the controls, outputs, and measurements vectors rather than scalars.

The form of the controller to be used for the design depends on its function. For this design, the controller must give the pilot a command capability. From Section 2.4, we recall that the command capability comes from the feedforward matrix which decouples the desired outputs. Appendix D shows that this matrix results from inverting the steady-state dynamic system. In flight, this matrix would exactly decouple the two desired outputs only if the helicopter were actually linear and had the same parameters as the model. Of course, this can never happen for several reasons:

- The helicopter dynamics are not linear.
- The helicopter has dynamics which were not modeled.
- The atmosphere is never "standard".
- The helicopter burns off fuel during flight, changing its inertia properties.
- The sensors have biases and scale factor errors.

The list could go on but it is clear we need to correct for differences between the model and the actual aircraft. One obvious solution is to use integral-error control. Figure 3.3 shows a form of the controller which includes this capability.

An interesting difference between this type of controller and more conventional flight control systems is the presence of off-diagonal terms in the integral-error feedback. Normally, the integral feedback is from desired output to the primary actuator for that output. For instance, a classically designed integral controller would use integral of velocity error for pitch control and integral of $\dot{h}$ error for collective. These off diagonal gains would be difficult to find using classical techniques since four transfer functions would be used to calculate the four integral gains. Finding these gains is straightforward using the RSANDY program. An-

$$\begin{bmatrix} \dot{u}(ft/sec) \\ \dot{w}(ft/sec) \\ \dot{q}(radians/sec) \\ \dot{\theta}(radians) \end{bmatrix} = F \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix} + G \begin{bmatrix} \delta_e(inches\ of\ longitudinal\ stick) \\ \delta_c(inches\ of\ collective\ lever) \end{bmatrix}$$

$$y_m(measurements) = \begin{bmatrix} q \\ \theta \\ \dot{h} \\ u \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -1.0 & 0 & 100.0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix} \tag{3.1}$$

Figure 3.2: **CH-47 Longitudinal Model.** This is a conventional longitudinal model found in most textbooks. The climb rate measurement is based on the approximation, $\dot{h} = -w + u_{nominal}\theta$.

the need for a fairly rigid control structure. This was necessary since the controller was to be programmed in fixed point assembly language on the aircraft's Sperry 1819A flight control computer. Since major controller structural changes would be difficult and time consuming, the baseline controller had to have a broad control structure which would allow design flexibility to come from changes in the compensator order or in the matrices themselves. With these goals and constraints in mind, the application of the methodology began.

3.2. Linear Models and Basic Control Structure

The models used for these designs come from Reference 11. This reference gives the aerodynamic coefficients at several different flight conditions and a general form of the "F" and "G" matrices based on these coefficients, the inertia properties of the aircraft, and the nominal airspeed. Appendix G shows this general form as well as the resulting linear models used for this research. Figure 3.2 shows the parameters of the longitudinal model.

Measurement	Units	Approx. Noise Standard Deviation	Approx. Correlation Time in Seconds	Spectral Density $S.D. \approx 2(\sigma^2)T_c$
$\theta_{vertical\ gyro}$	*degrees*	.1	.053	1.061×10^{-3}
$\phi_{vertical\ gyro}$	*degrees*	.1	.053	1.061×10^{-3}
$\psi_{directional\ gyro}$	*degrees*	1.0	.053	.1061
θ_{INS}	*degrees*	.05	.053	2.653×10^{-4}
ϕ_{INS}	*degrees*	.05	.053	2.653×10^{-4}
ψ_{INS}	*degrees*	.1	.053	1.061×10^{-3}
p	*degrees/sec*	2.0	.053	.424
q	*degrees/sec*	1.0	.053	.106
r	*degrees/sec*	.5	.053	2.653×10^{-2}
a_x	ft/sec^2	3.22	.053	1.1
a_y	ft/sec^2	3.22	.053	1.1
a_z	ft/sec^2	3.22	.053	1.1
$h_{barometric}$	ft	2.0	.318	2.546
h_{radar}	ft	.5	.053	2.653×10^{-2}
$vel_{pitot\ static}$	ft/sec	3.0	.318	5.73
$\alpha_{boom\ vane}$	*degrees*	2.0	.053	.424
$\beta_{boom\ vane}$	*degrees*	2.0	.053	.424

Figure 3.1: **Sensors Available on the NASA CH-47.** The standard deviations are estimates based on flight test data from the CH-47 instrumentation system. The .053 second correlation times corresponds to a 3 hz bandwidth.

Similarly, to establish a constant speed climb, the pilot must add collective thrust to begin climbing but simultaneously adjust pitch attitude to hold airspeed. In a sense, the pilot (through his extensive training) becomes an inner-loop decoupling controller, needed to give good speed and climb rate performance, the ultimate goal in cruising flight. The design goal, in terms of human factors, can be restated as reducing the pilot's workload in cruise flight by taking him out of the aircraft inner loops and giving him a direct velocity command system.

With the design goal stated, the physical hardware and flight software constraints must also be considered. Appendix F briefly describes the highly modified CH-47B flight research helicopter flown and maintained at the NASA Ames Research Center. The Langley Report gives a more detailed description of this particular helicopter, which has been modified to include a full authority fly-by-wire flight control system and extensive instrumentation.[10] Figure 3.1 shows the list of sensors available on the aircraft. Unfortunately, many of these sensors were quite noisy in the sense that they contained frequencies associated with the rotor motion. Rather than attempt to use the noisy measurements directly in the digital compensator, the TR-48 analog computer on the aircraft was programmed to filter several of the important aircraft motion sensor outputs. Specifically, fourth order Bessel filters were used on the longitudinal, lateral, and vertical accelerometers; a third order Bessel filter was used on the roll rate gyro; and second order Bessel filters on pitch rate, yaw rate, altitude, and pitot-static airspeed measurements. The filters were designed with 5 Hertz breakpoints to eliminate the "3 per rev" and "6 per rev" main rotor harmonics at 11 and 22 Hz. A brief description of these filters is included in Appendix H. A further constraint on the design process was

Chapter 3.

Longtidinal CAS for the CH-47

3.1. Design Goals and Constraints

The first example of an application of the methodology in Chapter 2 is the design of a longitudinal cruise autopilot for the CH-47. The goal was to synthesize a controller which gave the pilot independent control of airspeed and climb rate using separate pilot controls. For this design, the pilot longitudinal stick was used for the airspeed control and his collective lever was used for climb rate control. This implementation is unconventional since most helicopter control systems, even the highly augmented ones, give the pilot either pitch rate or pitch angle command from longitudinal stick and direct collective thrust command from the collective lever. Some of the more modern helicopters (CH-47D, HH60, CH-53) do close outer loops such as altitude or speed-hold around the collective thrust and pitch rate inner loops, but these are usually implemented as additional pilot-selectable autopilot modes. One trouble with pitch-angle, collective-thrust controllers is the coupling between the two. To increase speed in level flight, the pilot must pitch down, add collective thrust to maintain altitude and accelerate, then pitch up and reduce to a thrust slightly higher than the original to hold the new airspeed.

the compensator and then further simplifies the discrete compensator by putting it back into block minimal form. The SETPNT program uses an algorithm which finds the exact digital representation of a 2×2 compensator subsystem. One advantage of having the digital compensator in block minimal form is the reduction of arithmetic operations required by the computer. Specifically, the minimal form reduces the number of multiplies and adds, required by the compensator, by $r(r-1)$ where r is the order of the compensator. For large order compensators, such as required by systems controlling structural modes, the reduction can be important. The SETPNT program calculates this digital compensator in a form immediately usable in a floating point digital controller.

2.7. Summary

This chapter has shown an approach to designing control systems based on "modern control" methodologies. The approach is especially useful for MIMO systems of large order where compensator order reduction is essential. The Navion autopilot design was a simple example chosen to illustrate the design methodology. In applying this methodology to real control systems, problems will surface requiring the designer to modify and iterate the design to achieve the desired performance specifications. The next two chapters show this methodology applied to a real design problem. They illustrate the usefulness of the approach applied to a fairly complicated design task. They also show the type of problems which surface in real design applications.

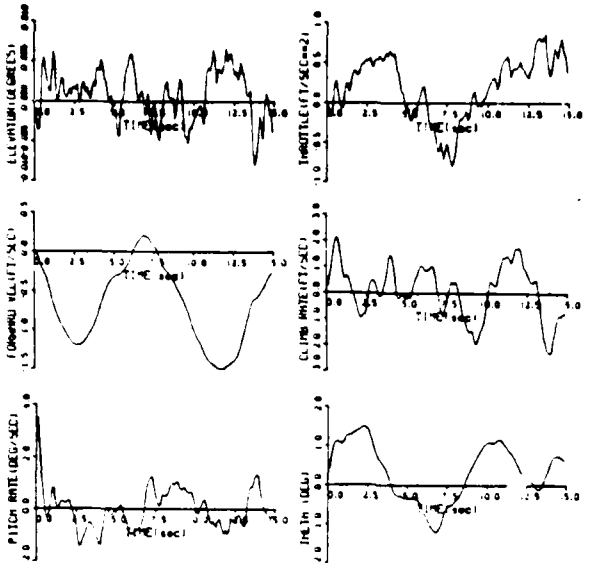


Figure 2.12: **Navion Reduced Order Compensator Performance in Turbulence.** The performance is nearly identical to the full order compensator (Figure 2.11) and is shown numerically in Figure 2.5

could be "tuned" by changing the weighting matrices and disturbance properties in the RSANDY program.

Eventually, the control system may be put into the physical system and tested. Since these compensator designs can be computationally intensive, it is unlikely to see any analog or continuous implementations for anything other than the simplest designs. Since the entire design process has been in the continuous domain, the final compensator must therefore be discretized to be useful in a digital computer based control system. The process of discretization is simple since the compensator is in a block minimal form. If the control is assumed to be a zero order hold (ZOH), i.e. step commands over the cycle time, then the discrete form can be solved exactly for the 2×2 and 1×1 blocks which make up the compensator dynamic system. The SETPNT program does this exact ZOH discretization for

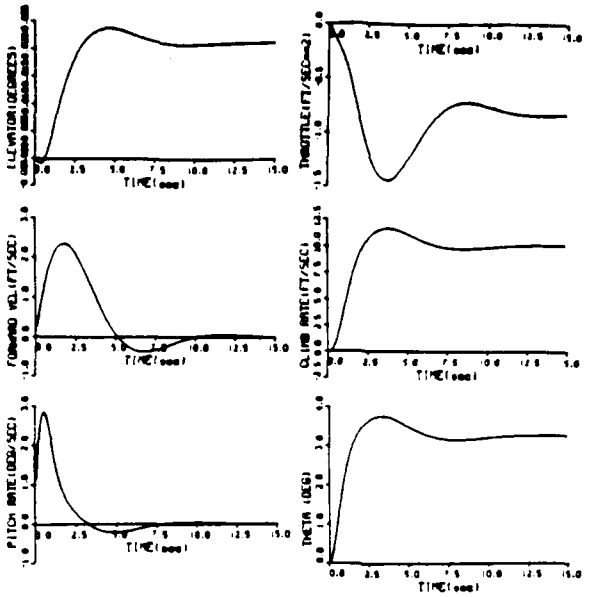


Figure 2.10: **Climb Rate Response for Reduced Order Compensator.** As with the velocity responses of Figures 2.7 and 2.8, the reduced order compensator has slightly reduced stability with a damping of about .6.

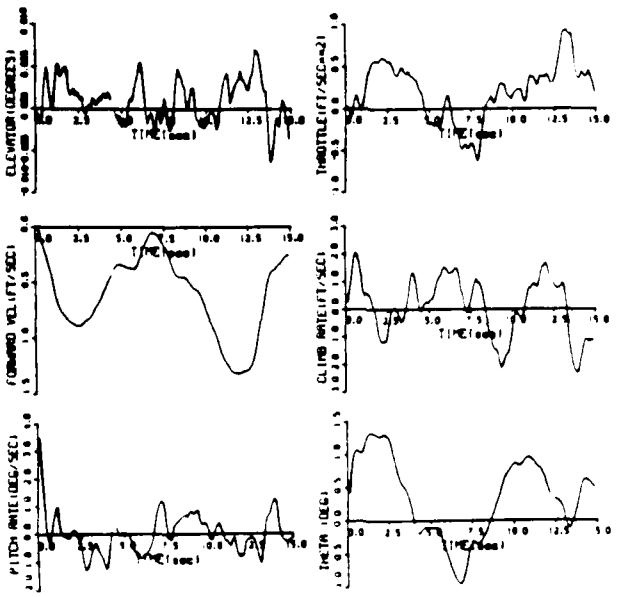


Figure 2.11: **Navion Full Order Compensator Performance in Turbulence.** The primary difference between the full order performance and reduced order performance (this figure and Figure 2.12) is in the control response which is smoother here.

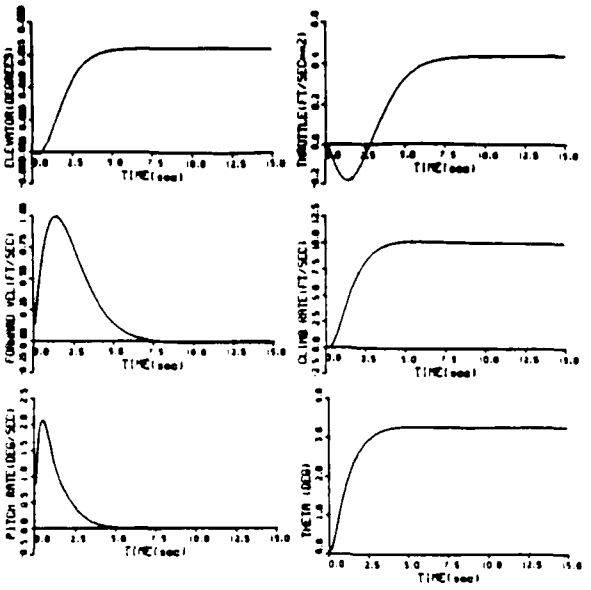


Figure 2.9: **Climb Rate Response for Full Order Compensator.** The rise time is 2 seconds with critical damping.

as we might expect. The velocity is better damped and faster using the full order compensator. Also, with the elimination of the θ measurement, we note the reduced order compensator has greater pitch angle excursions due to the velocity command. Comparing the climb rate responses of Figures 2.9 and 2.10, we note similar performance between the full and reduced order compensators. To compare the disturbance rejection characteristics of the two designs, we use the SIMPLOT program with plant disturbances and measurement noise. Figures 2.11 and 2.12 show the system performance in response to an initial pitch rate of 5 degrees in the presence of identical disturbances. As expected the full order compensator does a better job, as seen earlier in Figure 2.5. At this point, the design could be iterated by eliminating more modes or measurements, or by adding modes or measurements to improve performance. Alternately, the reduced order design

the responses when the commands *are* correctly accounted for in the compensator.

$$\dot{z} = A\,z + B\,y_m + T_{min}^{-1}\,T_{mod}^{-1}\,G\,u_{command}$$
$$u_{regulator} = C\,z \tag{3.9}$$

where:

- A: the block minimal form of the compensator dynamics matrix($F - GC - KH$) which already includes the regulator control effects (the GC term)
- B : The minimal form of the Kalman gain matrix from ROPTSYS
- C : The minimal form of the optimal regulator gain matrix from ROPTSYS
- G: the control distribution matrix for the physical system
- T_{min}^{-1}: the similarity transformation from the modal form to the block minimal form described in Appendix C
- T_{mod}^{-1}: the similarity transformation from the nominal form to the modal form, composed of the eigenvectors of $F - GC - KH$

At this point in the process, alternative approaches to the design were used.(Still before we realized how to implement Equation 3.9 above.) Two such alternatives are shown below:

- Pick noise spectral density matrices independent of the actual noise but which give good time responses.
- Use an inverse optimal solution as described by Bernard to establish the performance index weighting matrices and noise spectral density matrices.[7]

3.4.1. Redesign Using Arbitrary Measurement Spectral Densities

The first intermediate approach to improving the poor time response in the velocity channel required selecting noise spectral densities which give an adequate time response. This approach violates the assumptions of the LQG estimator design procedure where the Kalman filter gains are calculated to minimize the estimate errors in the presence of the given noise. However, the noise properties are

$$F_{min} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -260.43 & -20.31 & 0 & 0 & 0 & 0 \\ 0 & 0 & -14.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -27.3 & -6.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & -4.86 \end{bmatrix} \tag{3.10}$$

$$K_{min}(scaled) = \begin{bmatrix} -.58 & -12.1 & -.011 & .0078 \\ 6.98 & 226.5 & .0411 & .058 \\ -.72 & -100.7 & -.154 & .233 \\ -1.99 & -8.5 & .355 & -.75 \\ 9.12 & -3.78 & -2.36 & 4.35 \\ -11.34 & -49.3 & 2.44 & -4.44 \end{bmatrix} \tag{3.11}$$

$$K_{min}(unscaled) = \begin{bmatrix} -58 & -121 & -.011 & .0078 \\ 69.8 & 22649 & .0411 & .058 \\ -72 & -10070 & -.154 & .233 \\ -199 & -850 & .355 & -.75 \\ 912 & -378 & -2.36 & 4.35 \\ -1134 & -4930 & 2.44 & -4.44 \end{bmatrix} \tag{3.12}$$

$$C_{min}(scaled) = \begin{bmatrix} 4.05 & .74 & .75 & -.506 & -.46 & -.25 \\ 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix} \tag{3.13}$$

$$C_{min}(unscaled) = \begin{bmatrix} .405 & .074 & .075 & -.0506 & -.046 & -.025 \\ 0 & -.1 & -.1 & 0 & -.1 & -.1 \end{bmatrix} \tag{3.14}$$

System	Eigenvalues	Damping
Open Loop(F)	$-40.0, -40.0, -2.54, .503, -.105 \pm j.276$	$1, 1, 1, unstable, .35$
Regulator(F-GC)	$-4.64 \pm 5.82, -5.04 \pm 1.2, -2.94 \pm 4.01$	.62, .98, .59
Estimator(F-KH)	$-40.0, -40.0, -11.55 \pm 8.69, -3.74, -.0088$	1, 1, .79, 1, 1
Compensator(F-GC-KH)	$-10.15 \pm 12.54, -14.5, -3.39 \pm 3.97, -4.86$	.63, 1, .65, 1

Mode	Real	Imag	M_1	M_2
1	-10.15	12.54	277.9	27.4
2	-10.15	-12.54	-	-
3	-14.54	0	126	8.66
4	-3.4	3.97	68	20
5	-3.4	-3.97	-	-
6	-4.87	0	52	10.8

Figure 3.6: **CH-47 Longitudinal Full Order Compensator.** Notice the highly unstable open loop mode at $s = .503$. According to the test pilots, the 60 knot flight condition is the most unstable in the envelope.

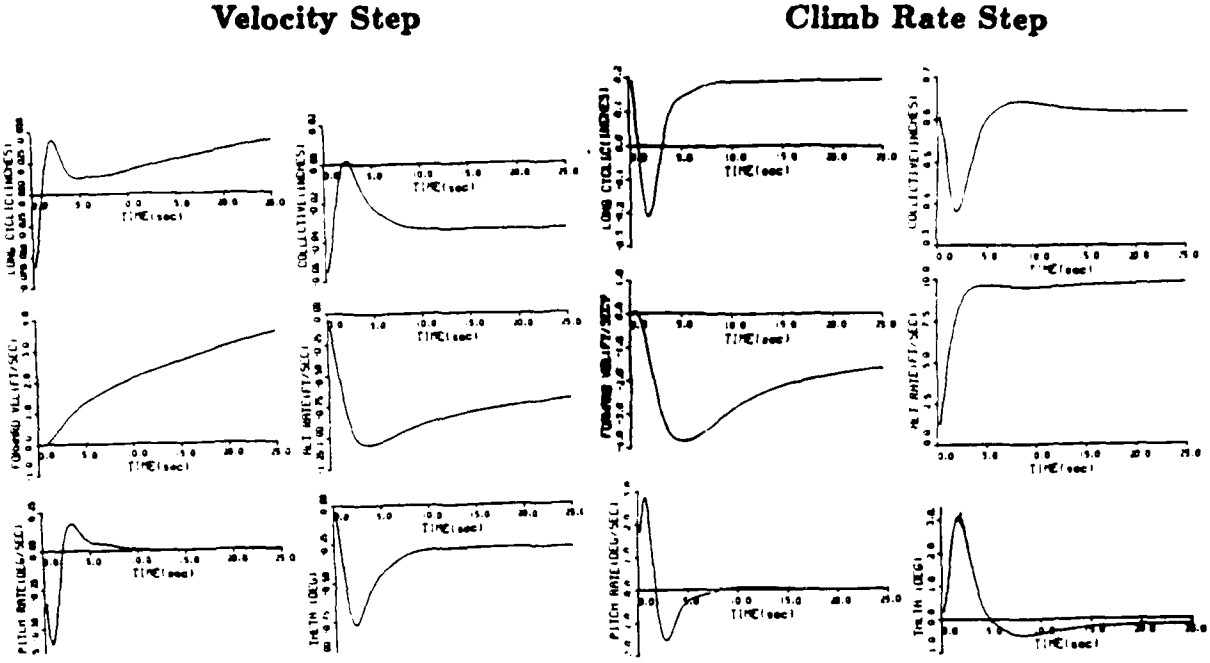


Figure 3.7: **Longitudinal CAS Time Responses using Nomimal Noise.** The poor velocity command response is dominated by the slow estimator mode ($s = -.0088$).

fairly uncertain and furthermore, the goal of this design was not a good estimator, but a good compensator. The selection of Q and R matrices which give "good" performance is an iterative process. Figure 3.9 shows the compensator when the spectral densities of all the measurements are set to .2. The very slow estimator pole is eliminated. Figure 3.10 shows the improved velocity response and the unchanged climb rate response.

3.4.2. Redesign Using an Inverse Optimal Controller

The second intermediate approach uses Bernard's "inverse optimal controller" technique.[7] This technique computes A, B, Q and R matrices which, if used in an LQG design, give a closed loop system with desired poles. Figure 3.11 shows this compensator including the A, B, Q and R provided by Bernard. The time

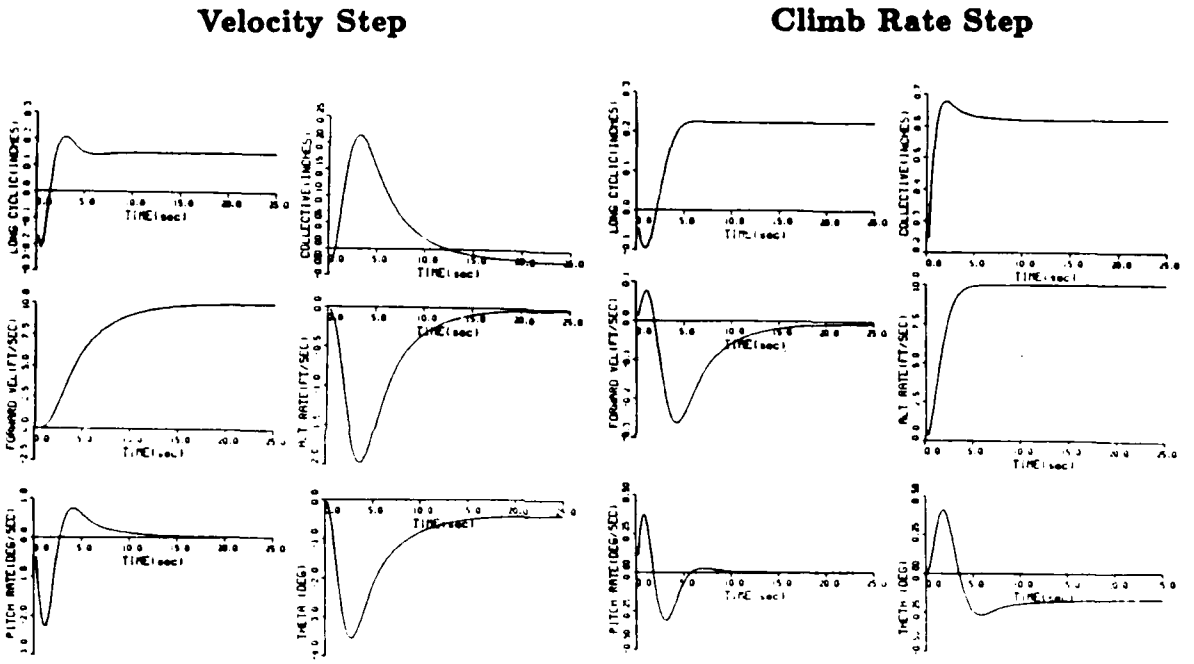


Figure 3.8: **Longitudinal CAS Time Responses with Commands to the Compensator.** With the command correctly fed to the compensator, the response is identical to a full-state feedback controller. The climb rate response is unchanged from Figure 3.7.

$$F_{min} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -14.715 & -7.43 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -4.27 & -1.61 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -2.7 & -3.16 \end{bmatrix} \tag{3.15}$$

$$K_{min}(scaled) = \begin{bmatrix} .00014 & .00048 & -.009 & -1.12 \\ -.001 & -.0018 & -2.40 & 3.99 \\ -.0076 & -.00093 & -.18 & .99 \\ .0012 & .0018 & 3.11 & -3.66 \\ -.000093 & 1.1 \times 10^{-4} & -.15 & -.24 \\ .00015 & .0018 & -.026 & -.20 \end{bmatrix} \tag{3.16}$$

$$K_{min}(unscaled) = \begin{bmatrix} .014 & .048 & -.009 & -1.12 \\ -.10 & -.18 & -2.40 & 3.99 \\ -.076 & -.093 & -.18 & .99 \\ .12 & .18 & 3.11 & -3.66 \\ -.0093 & .00011 & -.15 & -.24 \\ .015 & .018 & -.026 & -.20 \end{bmatrix} \tag{3.17}$$

$$C_{min}(scaled) = \begin{bmatrix} 0.0 & 1.0 & 0.0 & 1.0 & 1.65 & 1.17 \\ .26 & .40 & .23 & .38 & 0.0 & 1.0 \end{bmatrix} \tag{3.18}$$

$$C_{min}(unscaled) = \begin{bmatrix} 0.0 & -.1 & 0.0 & -.1 & -.165 & -.117 \\ -.026 & -.040 & -.023 & -.038 & 0.0 & -.1 \end{bmatrix} \tag{3.19}$$

System	Eigenvalues	Damping
Open Loop(F)	$-40.0, -40.0, -2.54, .503, -.105 \pm j.276$	$1, 1, 1, unstable, .35$
Regulator(F-GC)	$-2.62, -.866 \pm j.913, -1.035 \pm .384, -.281$	$1, .69, .94, 1$
Estimator(F-KH)	$-40.0, -40.0, -4.45, -1.0 \pm j.721, -1.31$	$1, 1, 1, .81, 1$
Compensator(F-GC-KH)	$-3.72 \pm j.947, -.803 \pm j1.90, -1.58 \pm j.445$	$.97, .39, .96$

Mode	Real	Imag	M_1	M_2
1	-3.72	.947	11.33	3.05
2	-3.72	-.947	–	–
3	-.803	1.90	5.24	6.53
4	-.803	-1.90	–	–
5	-1.58	.45	.254	1.61
6	-1.58	-.45	–	–

Figure 3.9: **Longitudinal CAS using Arbitrary Measurement Spectral Density.** The slow estimator mode of Figure 3.6 is eliminated.

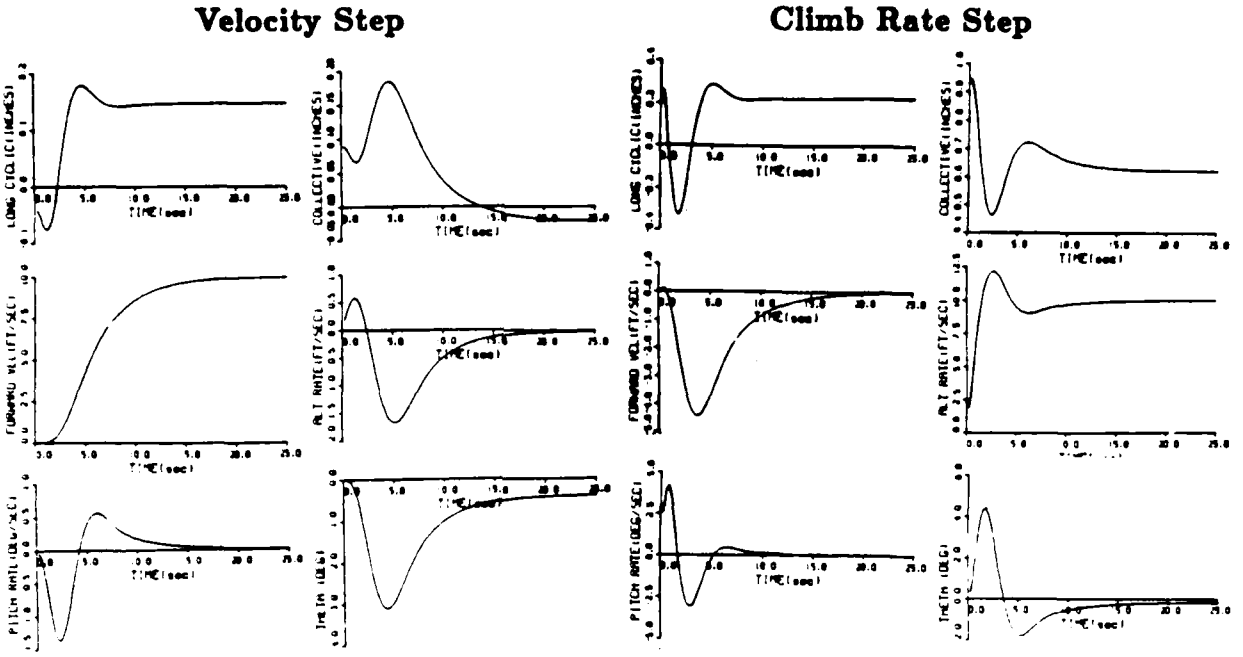


Figure 3.10: **Longitudinal CAS Time Responses using Arbitrary Noise.** The poor velocity response of Figure 3.7 is eliminated.

responses of Figure 3.12 confirm the improved performance.

Figure 3.13 shows the statistical performance of the original design and the two alternate designs. As expected, the nominal design has the lowest errors. The inverse optimal controller is nearly as good but the other is clearly the worst. Based on the combination of statistical performance and the time responses, the inverse optimal design was the best of the full order compensators. With the full order compensator in hand, the process of order reduction began.

3.5. Compensator Order Reduction

Using the M_2 criterion of section 2.3, both the nominal and the inverse optimal compensators indicate reduction to third order is feasible. From a more practical

$$F_{min} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -205.73 & -22.68 & 0 & 0 & 0 & 0 \\ 0 & 0 & -3.73 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1.55 & -2.33 & 0 \\ 0 & 0 & 0 & 0 & 0 & -.68 \end{bmatrix} \tag{3.20}$$

$$K_{min}(scaled) = \begin{bmatrix} -.013 & -.87 & -.34 & -.0015 \\ -.17 & 4.95 & .17 & -1.03 \\ .37 & -.84 & .01 & 1.84 \\ -.011 & -.43 & -.12 & .88 \\ .022 & .34 & .069 & -.57 \\ .014 & .63 & .16 & -1.40 \end{bmatrix} \tag{3.21}$$

$$K_{min}(unscaled) = \begin{bmatrix} -1.29 & -86.68 & -.033 & -.0015 \\ -16.62 & 494.75 & .17 & -1.03 \\ 37.28 & -84.30 & .01 & 1.84 \\ -1.132 & -42.52 & -.12 & .88 \\ 2.16 & 33.85 & .069 & -.57 \\ 1.44 & 63.12 & .16 & -1.40 \end{bmatrix} \tag{3.22}$$

$$C_{min}(scaled) = \begin{bmatrix} 0.0 & 1.0 & 1.0 & 1.61 & .33 & 1.0 \\ .10 & .25 & .21 & 0.0 & 1.0 & -.42 \end{bmatrix} \tag{3.23}$$

$$C_{min}(unscaled) = \begin{bmatrix} 0.0 & -0.1 & -.1 & -.16 & -.033 & -.1 \\ -.01 & -.025 & -.021 & 0.0 & -.1 & .042 \end{bmatrix} \tag{3.24}$$

System	Eigenvalues	Damping
Open Loop(F)	$-40.0, -40.0, -2.54, .503, -.105 \pm j.276$	$1, 1, 1, unstable, .35$
Regulator(F-GC)	$-2.13 \pm j.78, -.56 \pm j.23, -.82 \pm j.33$	.94, .93, .93
Estimator(F-KH)	$-40.0, -40.0, -11.54 \pm j8.70, -1.01, -.52$	1, 1, ..80, 1, 1
Compensator(F-GC-KH)	$-11.34 \pm j8.78, -3.73, -1.17 \pm j.44, -.68$	.79, 1, .94, 1

Mode	Real	Imag	M_1	M_2
1	-11.34	8.78	15.33	1.35
2	-11.34	-8.78	–	–
3	-3.73	0.0	2.11	.56
4	-1.17	.44	2.59	2.22
5	-1.17	-.44	–	–
6	-.68	0.0	1.67	2.46

Figure 3.11: **Longitudinal Compensator based on Inverse Optimal Solution.** The slow estimator mode of the nominal design, Figure 3.6, is gone.

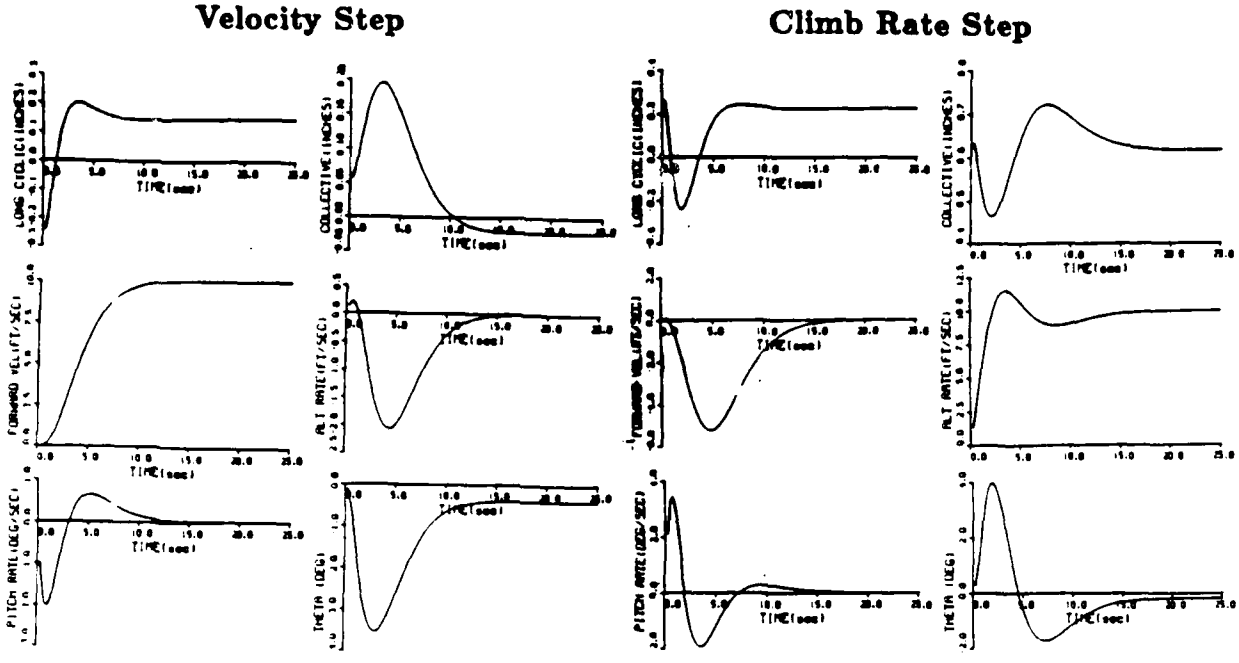


Figure 3.12: **Longitudinal CAS Time Responses using Inverse Optimal Controller.** The velocity reponse is much improved from Figure 3.7.

System	Standard Deviation						
	δ_e (inches)	δ_e (inches)	u (ft/sec)	w (ft/sec)	q (deg/sec)	θ (deg)	$\dot{h}$ (ft/sec)
Nominal	.12	.06	1.01	1.51	0.84	0.78	1.49
Arbitrary Noise	.39	.21	4.24	3.96	3.46	3.34	2.65
Inverse Optimal	.16	.07	1.43	2.90	1.24	1.29	1.72

Figure 3.13: **Performance Comparison of Full Order Compensators.** These data are based on the measurement noise shown in Figure 3.1 and disturbance noise of Figure 3.4.

standpoint, using output feedback of the measurements is also a possibility. This is appropriate if we recall from section 3.1 that all the measurements were being analog filtered before entering the flight control computer. The next two sections show these two designs.

3.5.1. Robust Longitudinal Third Order Compensator

Using the nominal compensator reduced to third order as a starting guess, the RSANDY program was used to optimize the low order compensator using its robust design feature. The following flight conditions were used with the weightings shown.

Condition	Airspeed(knots)	Climb Rate(ft/min)	Weighting
Nominal	60	0	.8
Off Nominal 1	60	500	.05
Off Nominal 2	60	-500	.05
Off Nominal 3	40	0	.05
Off Nominal 4	80	0	.05

Stability is ensured by the RSANDY program for each of these flight conditions. Figure 3.14 shows the results including the closed loop roots at the nominal flight condition. There is still a slow mode ($s = -.059$) which is shown in the time responses of Figure 3.15 indicating the need for integral control.

When the integral control loop is added to the third order controller, the RSANDY program was again used to set the four gains in the K_I matrix of Figure 3.3. The program did not converge with only the four gains of K_I allowed to vary in the optimization. By releasing both the K_I *and* the compensator gains, the compensator of Figure 3.16 emerged. Figure 3.17 shows the associated time responses. The same approach was used to set the integral gains for the full-order compensator when the integral control loop was added. Figures 3.18 and 3.19

$$F_{min} = \begin{bmatrix} 0.0 & 1.0 & 0.0 \\ -226.3 & -78.49 & 0.0 \\ 0.0 & 0.0 & -1.397 \end{bmatrix} \tag{3.25}$$

$$K_{min} = \begin{bmatrix} -22.17 & -121.7 & -.35 & -.11 \\ 53.14 & 3527. & .87 & .24 \\ -31.38 & -84.46 & -.90 & -.38 \end{bmatrix} \tag{3.26}$$

$$C_{min} = \begin{bmatrix} 0.0 & -.10 & -.10 \\ -.11 & -.056 & .012 \end{bmatrix} \tag{3.27}$$

Real Part	Imag Part	Damping	Freq(rad/sec)	Freq(Hz)
-75.45	0.0	1.0	75.45	12.01
-40.00	0.0	1.0	40.00	6.37
-40.18	0.0	1.0	40.18	6.40
-3.12	0.0	1.0	3.12	.50
-.96	1.36	.58	1.67	.27
-.96	-1.36	.58	1.67	.27
-.059	0.0	1.0	.059	.0094
-.42	0.0	1.0	.42	.068
-.96	0.0	1.0	.96	.15

Figure 3.14: **Longitudinal Reduced Order Compensator.** Reducing from 6^{th} to 3^{rd} order decreases the number of independent gains in the compensator from 36 to 18.

Velocity Step **Climb Rate Step**

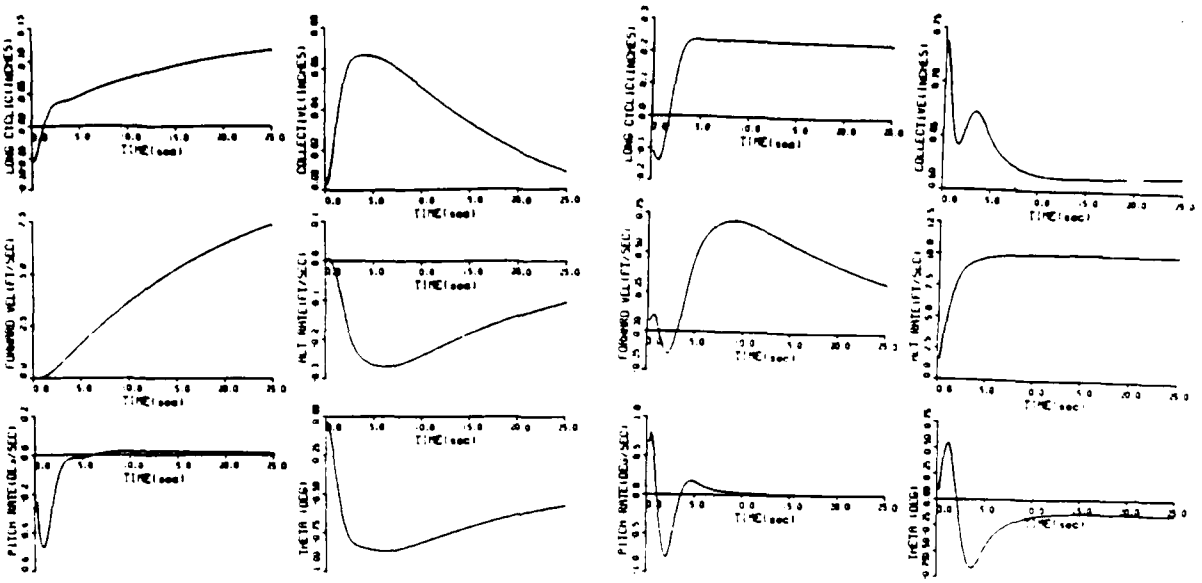


Figure 3.15: **Longitudinal Reduced Order Compensator Time Responses.** The velocity response is still quite slow but climb rate is adequate.

show the resulting compensator and time responses.

3.5.2. Longitudinal Output Feedback Controller

Since the measurements available ($q\theta u\dot{h}$) are nearly the same [2] as the states of a fourth order model, the gains of a regulator design are a good starting point in using RSANDY to compute output feedback gains. In fact, using these gains with no further optimization gave quite impressive performance as shown in Figures 3.20 and 3.21.

As with the reduced order compensator, an integral control loop is necessary to ensure that the actual commands are achieved in flight. Using the K_I gains from the reduced order design(Figure 3.16) and the output feedback controller of

[2] The fourth order model has states ($uwq\theta$) and $\dot{h} \approx u_{nominal}\theta - w$.

$$A_{min} = \begin{bmatrix} 0.0 & 1.0 & 0.0 \\ -229.5 & -16.82 & 0.0 \\ 0.0 & 0.0 & -2.48 \end{bmatrix} \tag{3.28}$$

$$K_{min} = \begin{bmatrix} -22.83 & -190.9 & -.07 & .059 \\ 57.00 & 3528. & .044 & -.33 \\ -18.57 & -59.93 & -1.37 & -.67 \end{bmatrix} \tag{3.29}$$

$$C_{min} = \begin{bmatrix} 0.0 & -.1 & -.1 \\ .45 & -.071 & .041 \end{bmatrix} \tag{3.30}$$

$$I = \begin{bmatrix} -.0054 & .0098 \\ -.00823 & .047 \end{bmatrix} \tag{3.31}$$

Closed Loop Eigenvalues

Real Part	Imag Part	Damping	Freq(rad/sec)	Freq(Hz)
-39.75	0.0	1.0	39.75	6.32
-40.03	0.0	1.0	40.03	6.37
-7.61	11.46	.55	13.75	2.19
-7.61	-11.46	.55	13.75	2.19
-1.81	2.17	.64	2.83	.45
-1.81	-2.17	.64	2.83	.45
-1.77	0.0	1.0	1.77	.28
-.044	.092	.44	.10	.016
-.044	-.092	.44	.10	.016
-.53	.37	.82	.65	.10
-.53	-.37	.82	.65	.10

Figure 3.16: **Longitudinal Reduced Order Compensator with Integral Control.** The A_{min}, K_{min}, and C_{min} matrices are not the same as in Figure 3.14.

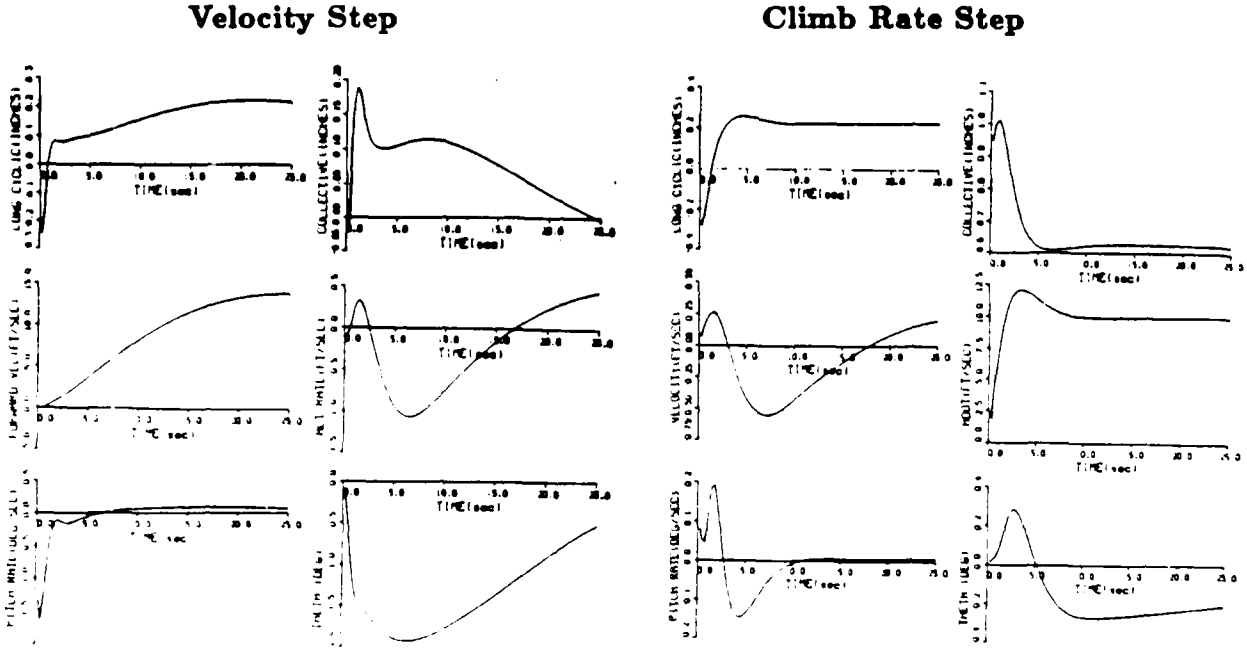


Figure 3.17: **Longitudinal Reduced Order Integral Controller Time Responses.** Although the velocity response is still fairly slow, this design was selected for evaluation in flight.

$$F_{min} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -205.7 & -22.67 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1.781 & -2.661 & 0 & 0 \\ 0 & 0 & 0 & 0 & -4.279 & 0 \\ 0 & 0 & 0 & 0 & 0 & -.263 \end{bmatrix} \tag{3.32}$$

$$K = \begin{bmatrix} -1.96 & -86.7 & -.148 & -.0213 \\ -16.62 & 494.8 & .166 & -.81 \\ -1.11 & -42.6 & -.437 & .475 \\ 2.171 & 33.8 & .727 & -.174 \\ 37.3 & -84.3 & .148 & 1.09 \\ 1.33 & 63.1 & .0823 & -.395 \end{bmatrix} \tag{3.33}$$

$$C = \begin{bmatrix} 4.932 & .122 & .477 & 1.01 & -.068 & -.44 \\ -.0105 & -.025 & 0 & -.1 & -.021 & .042 \end{bmatrix} \tag{3.34}$$

$$I = \begin{bmatrix} -.00003 & -.01 \\ .000157 & .053 \end{bmatrix} \tag{3.35}$$

Closed Loop Eigenvalues

Real Part	Imag Part	Damping	Freq(rad/sec)	Freq(Hz)
-40.01	.0022	1.0	40.01	6.37
-40.01	-.0022	1.0	40.01	6.37
-11.43	8.21	.81	14.07	2.239
-11.43	-8.21	.81	14.07	2.239
-.543	2.87	.185	2.93	.466
-.543	-2.87	.185	2.93	.466
-3.38	0.0	1.0	3.38	.537
-2.86	0.0	1.0	2.86	.455
-.299	.90	.315	.947	.151
-.299	-.90	.315	.947	.151
-.561	.334	.86	.65	.104
-.561	-.334	.86	.65	.104
-.206	0.0	1.0	.206	.0329
-4.24×10^{-5}	0.0	1.0	4.24×10^{-5}	6.75×10^{-6}

Figure 3.18: **Longitudinal Full Order Compensator with Integral Control.** As with the third order design, the A, K, and C matrices are different from the full order design without integral control (Figure 3.11).

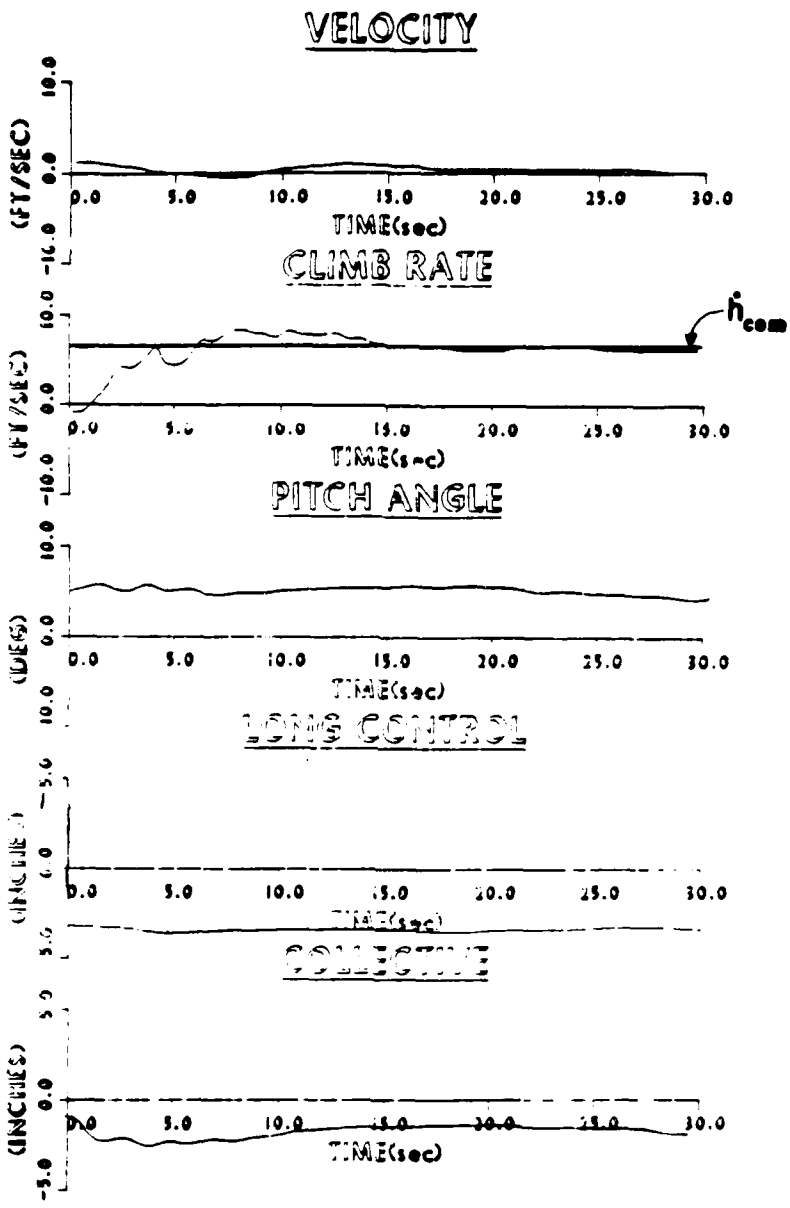


Figure 3.31: **Third Order Compensator Flight Response to Climb Rate Command.** Atmospheric turbulence masked the transient portion of the response but steady state was achieved in about 10 seconds, similar to the simulation of Figure 3.17.

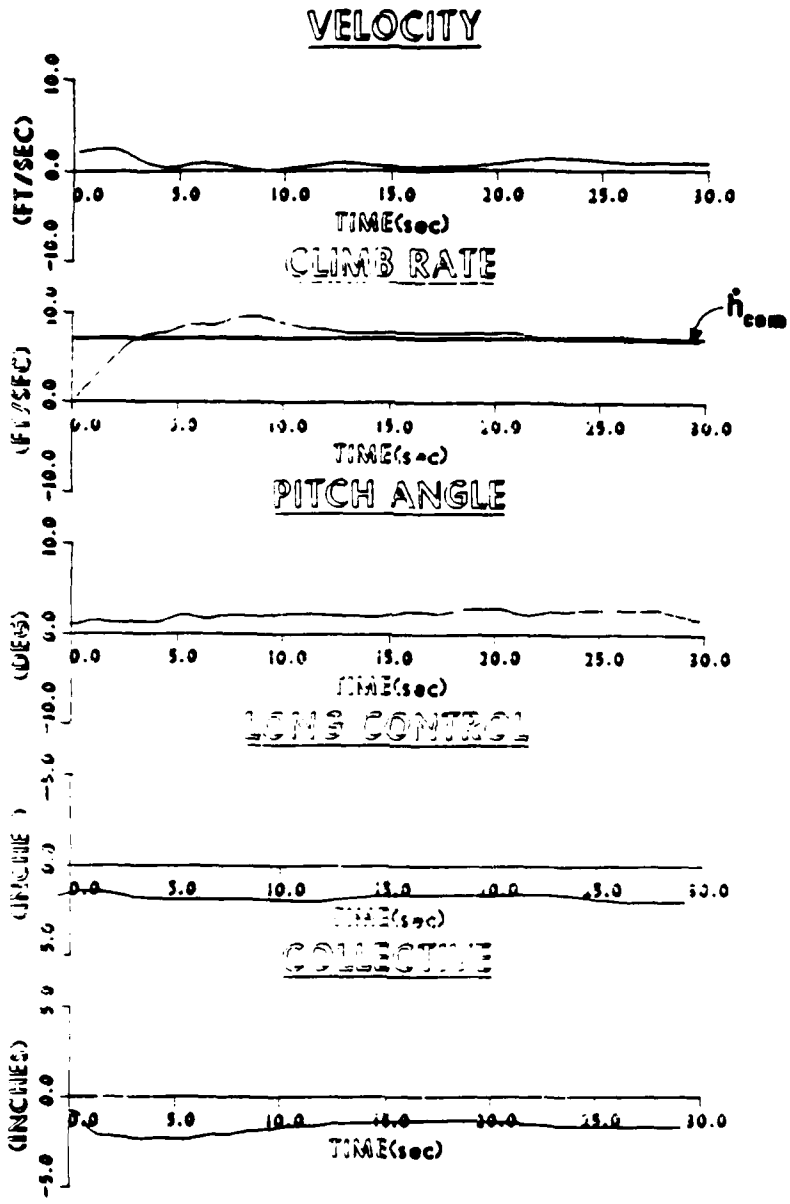


Figure 3.30: **Full Order Compensator Flight Response to Climb Rate Command.** As in the velocity response of Figure 3.27, the $\dot{h}$ command response was well predicted by the simulation results of Figure 3.19. This response also shows the decoupling between velocity and climb rate.

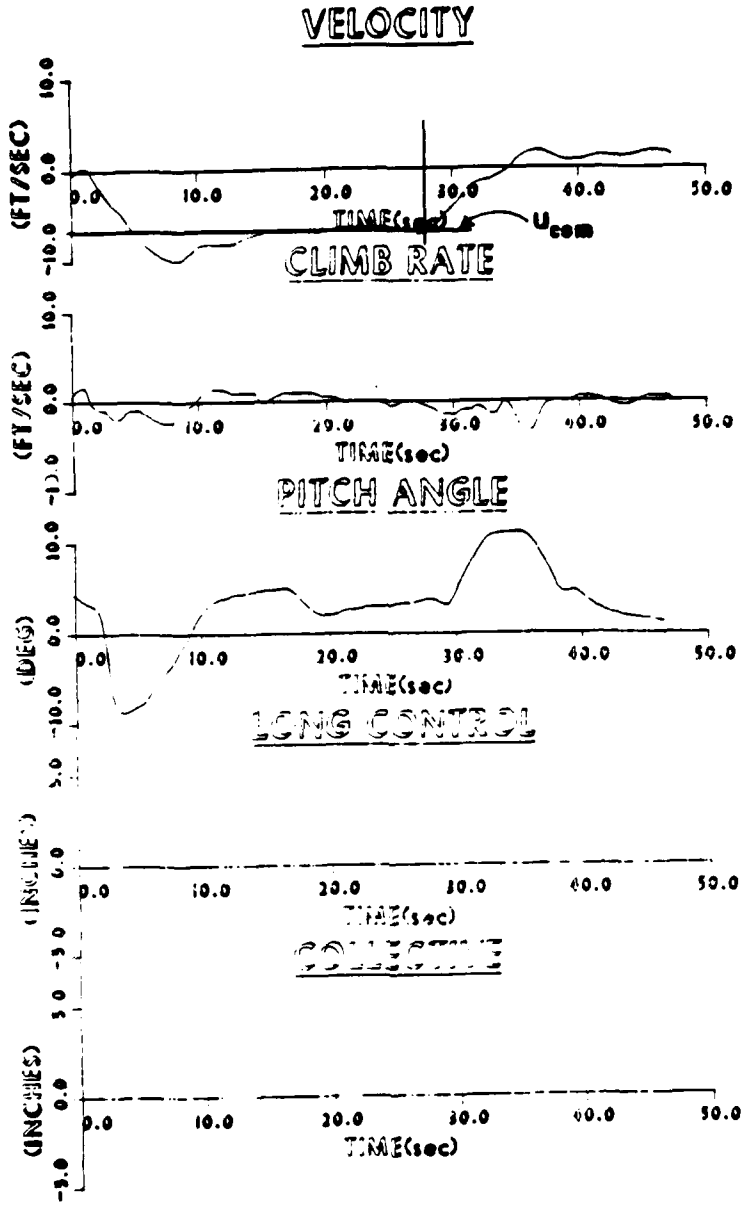


Figure 3.29: **Output Feedback Compensator Flight Response to Velocity Command.** As in the responses of Figures 3.27 and 3.28, this flight response agrees with the simulation predictions. In this case, however, the overall loop gain (final gain to the actuator) was reduced by half.

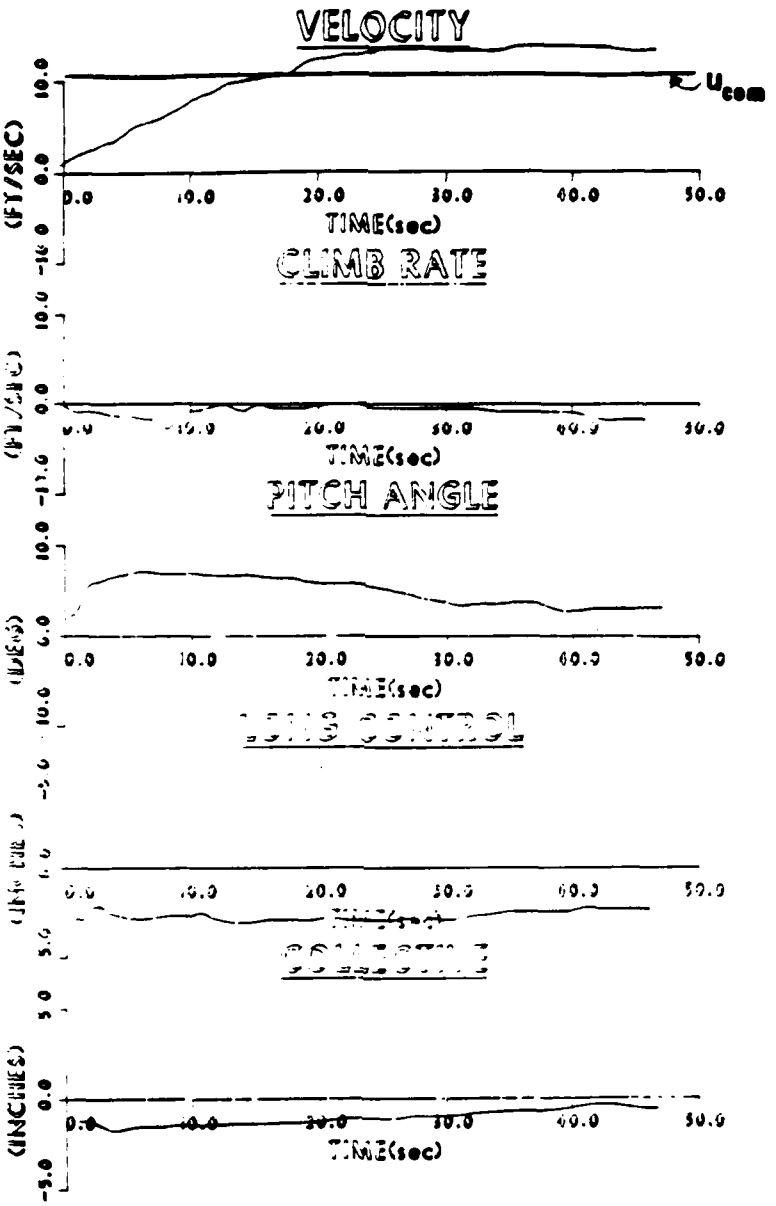


Figure 3.28: **Third Order Flight Response to Velocity Command.** This reponse to a command of approximately 10 ft/sec follows closely the simulation results of Figure 3.17.

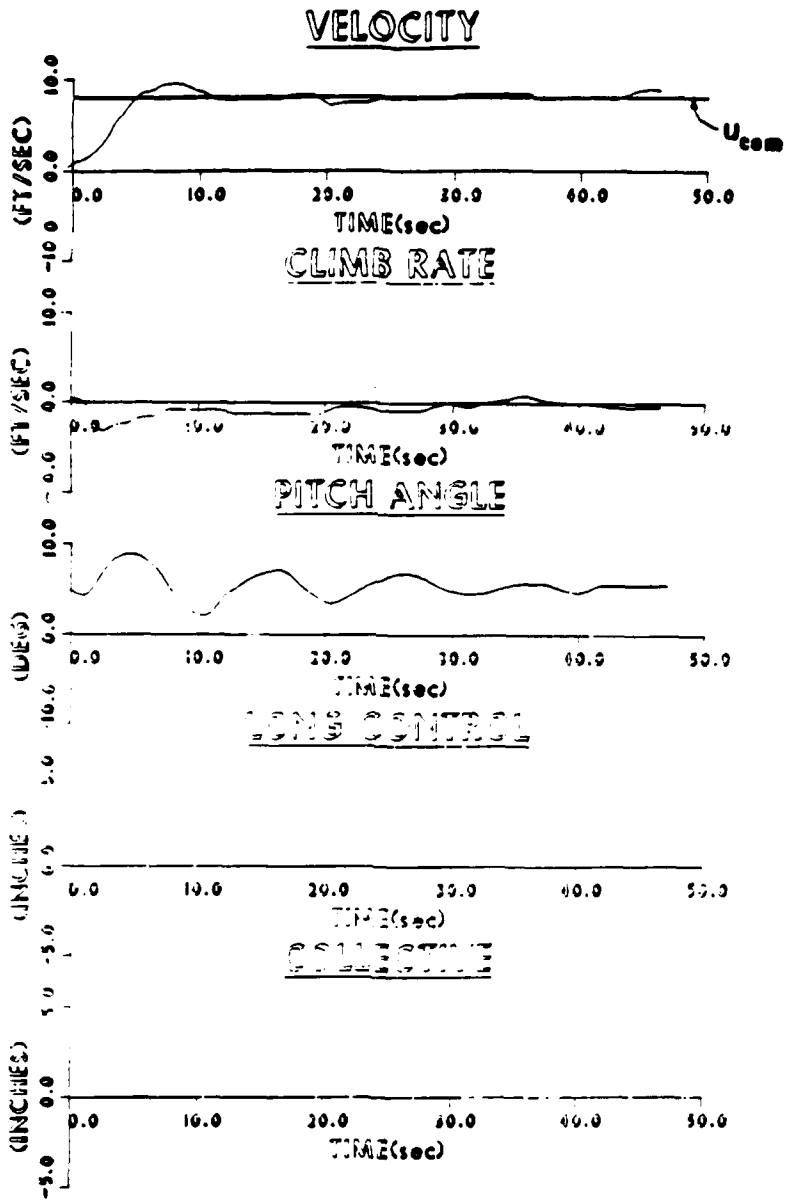


Figure 3.27: **Full Order Compensator Flight Response to Velocity Command.** The rapid velocity response and poorly damped pitch angle were also evident in the simulation response of Figure 3.19.

in a realistic task such as flying a precision approach.

3.8. Summary of Results for the Longitudinal CAS Design

The results of this section can be separated into two categories:

- the design task
- the flight test results

The design task was important because it established an experience base for use of the design methodology. This task showed that use of integral-error feedback in a state variable based controller eliminates the effects of an inaccurate plant model in achieving commanded outputs. It also showed the difficulty encountered when using LQG design techniques with specified time domain properties. In this example, three methods of setting weighting matrices and spectral density matrices were compared. From a practical standpoint, this first design task was very important since an accurate way of scaling the analytical design for use in the fixed point flight computer was developed. An equally important result of this task was the coding and testing of a general form of a modern controller for the Sperry 1819 flight computer.

The flight test results showed the technical feasibility of decoupled control using a "modern" controller. The test system had adequate handling qualities but, due to time constraints, no attempt was made to iterate the designs for "good" handling qualities. Also, the disturbance rejection capabilities were not thoroughly investigated.

took place at the Crows Landing test facility(in the San Joaquin valley) to avoid the heavy traffic of the South San Francisco Bay Area near Ames Research Center. The purpose of the flight test was to validate the performance of the different controllers rather than "tune" the system for maximum pilot acceptance. Step commands from the computer or the pilot were used to evaluate the different systems. The nominal airspeed was 60 knots but stability and performance were checked from 40 to 80 knots. Figures 3.27, 3.28 and 3.29 show the responses of the full-order, third-order, and direct feedback compensators to velocity step commands of about 10 ft/sec. All three of these systems show good decoupling between velocity and climb rate. Their velocity responses are similar to the simulation results of figures 3.12, 3.17, and 3.23 but the pitch angle behavior is less damped than the simulation results. Chen has shown that unmodeled dynamics, especially rotor dynamics, are the probable cause for the lower achievable control bandwidth in flight.[12] In fact, the output feedback flight implementation had the overall pitch and collective loop gains reduced by half to achieve an acceptable response. Figures 3.30, 3.31 and 3.32 show similar results for climb rate commands. The importance of integral control is demonstrated in figure 3.33 which shows the response to a climb rate command for the third order controller without integral control. Although a steady climb rate is achieved, an unwanted velocity change of similar magnitude is also present.

Pilot opinion was mixed concerning the system. They were impressed at how well it held airspeed and climb rate but the transient behavior was not totally acceptable and the system was "sloppy" in response to gusts. This is valid criticism based on the pitch angle responses. Unfortunately the system was not evaluated

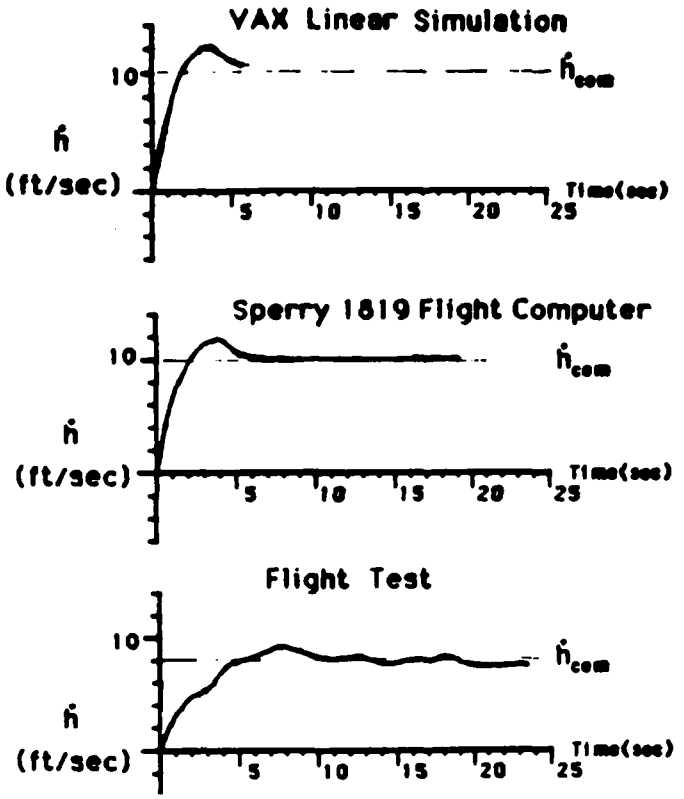


Figure 3.26: **Comparison of Analytical Design and Flight Test Implementation of the Controller.** This figure confirms the correctness of the steps needed to go from the analytical design to flight. These responses are for the 3^{rd} order compensator with integral control.

tinuous designs were digitized and scaled correctly. The OBS was also useful in preliminary pilot evaluation of the control laws and in initial setting of the stick sensitivities and other pilot-related items. Figure 3.26 shows a comparison of the time responses for a climb rate command for the third order system. This confirmed the accurate digitization and scaling of the analytical design. All flight controllers were similarly checked prior to each flight.

3.7. Flight Test Results

Once the designs had been implemented and checked out in the flight hardware, the flight testing began. Since the controller was designed for cruise, the testing

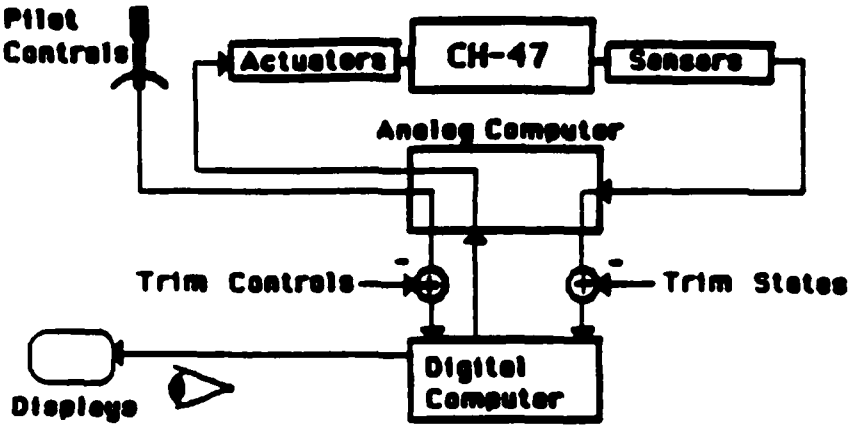


Figure 3.25: **Flight Test Implementation.** Sensor outputs and actuator commands were filtered in the analog computer while the control laws were executed in the digitial computer.

pose matrix multiply routine was programmed to take advantage of the minimal form of the compensator dynamics matrix.

- The existing instrumentation output subroutines were modified to send internal compensator data to the ground support station.

- Since the Sperry 1819A flight computer is an 18 bit fixed point machine, the matrices from the previous section were scaled to avoid numerical overflow during program execution. Appendix E describes this "fixed point scaling" technique and lists the SCALEM computer programs which accomplish this.

The actual assembly code implementing the controller is shown in Appendix I.

One important capability of the research system was onboard simulation(OBS). The OBS allowed the real-time flight software to be checked in the closed-loop system prior to actual flight. It was especially useful in confirming that the con-

could work without integral assistance. This would determine how well the design models compared with the real aircraft. Using the SETPNT program, the digital block minimal forms of each of these compensators were computed for flight implementation.

3.6. Flight Test Implementation

Once the analytical designs were complete, the tedious task of actual flight implementation began. Figure 3.25 shows a block diagram of the control structure on the research vehicle. The TR-48 analog computer was used to filter the sensor outputs and the digital actuator commands from the Sperry computer. The digital commands from the computer were filtered to avoid possible actuator wear caused by the 20 Hz chatter. The control laws were implemented in the Sperry 1819A digital flight control computer. Before the designs of the previous sections could be tested, the flight control computer software had to be modified to use them. In order to implement design changes more quickly, the flight software was set up to use the compensator matrices directly. Some of the considerations involved in programming the general form of the compensator in assembly language (shown in Figure 3.3) are listed below:

- Since the design was based on a linear perturbation model, the measurement and control trim values were approximated as the values at engage time; these were subtracted from their sensed values for use by the controller. This had the added advantage of eliminating engage transients.
- The compensator states were initialized to zero before system engage.
- To minimize time required for the compensator calculations, a special pur-

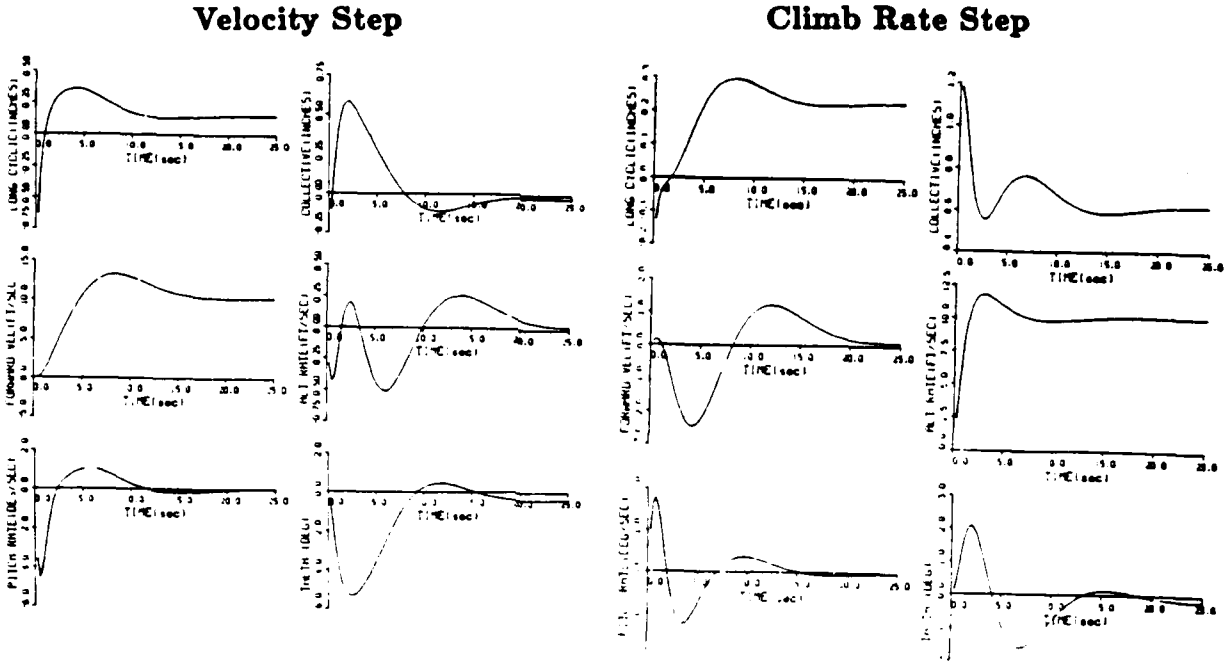


Figure 3.23: **Longitudinal Output and Integral Feedback Controller Time Responses.** This design was selected for flight evalualtion.

Standard Deviation							
System	δ_e (inches)	δ_c (inches)	u (ft/sec)	w (ft/sec)	q (deg/sec)	θ (deg)	h (ft/sec)
Nominal	.12	.06	1.01	1.51	0.84	0.78	1.49
Arbitrary Noise	.39	.21	4.24	3.96	3.46	3.34	2.65
Inverse Optimal	.16	.07	1.43	2.90	1.24	1.29	1.72
Inv Opt w/Int Cntrl	.34	.18	.82	1.53	2.0	1.1	1.64
3^{rd} **Order**	.08	.08	2.83	0.83	0.67	0.78	1.39
3^{rd} **Order w/ Int Cntrl**	.10	.14	0.72	0.83	0.55	0.38	0.86
Output Feedback	.32	.19	.82	1.58	1.29	0.80	0.80
Output Feedback w/Int Cntrl	.32	.21	.93	1.54	1.30	0.84	0.81

Figure 3.24: **Performance Comparison of All Compensators.** The boldbaced systems were selected for flight evaluation.

$$D = \begin{bmatrix} -5.74 & -12.51 & .041 & .103 \\ -1.23 & -10.44 & -.057 & .013 \end{bmatrix} \tag{3.37}$$

$$I = \begin{bmatrix} -.017 & .054 \\ -.013 & .084 \end{bmatrix} \tag{3.38}$$

Closed Loop Eigenvalues

Real Part	Imag Part	Damping	Freq(rad/sec)	Freq(Hz)
-37.52	0.0	1.0	37.52	5.97
-39.42	0.0	1.0	39.42	6.27
-2.39	0.0	1.0	2.39	.38
-1.17	.58	.89	1.30	.21
-1.17	-.58	.89	1.30	.21
-.13	0.0	1.0	.13	.021
-.23	.36	.54	.43	.068
-.23	-.36	.54	.42	.068

Figure 3.22: **Longitudinal Output and Integral Feedback Controller.** As with the full order and 3^{rd} order designs, the use of integral control reduces damping slightly.

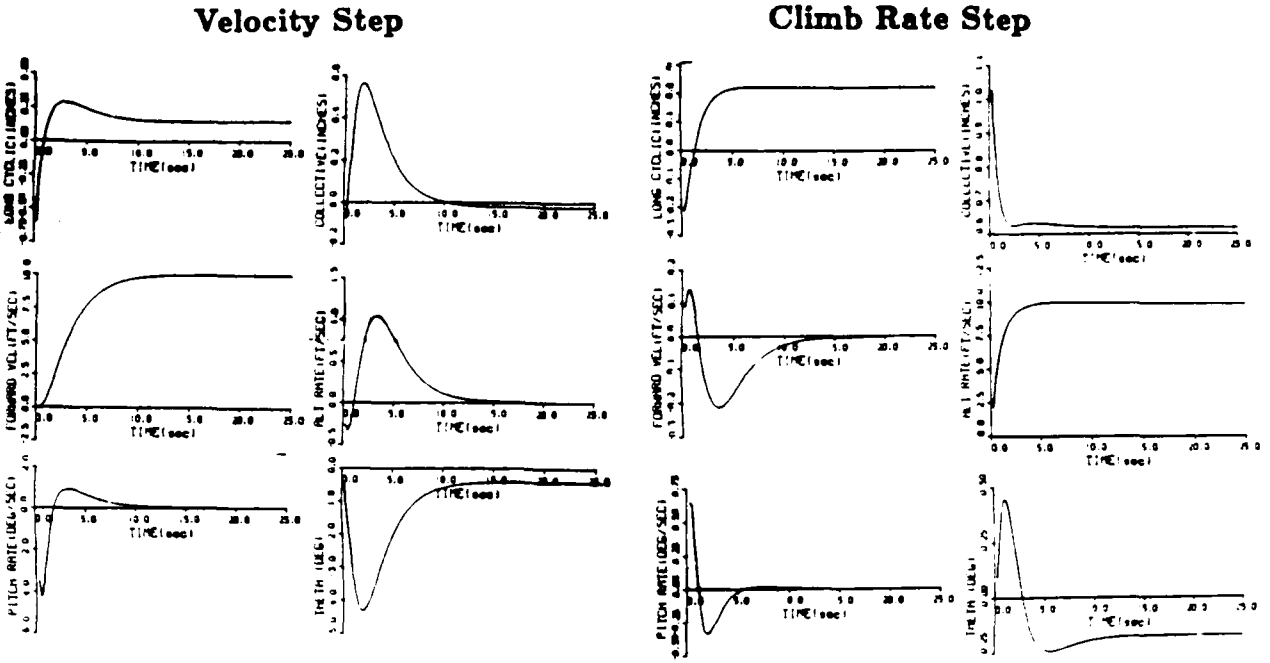


Figure 3.21: **Longitudinal Output Feedback Controller Time Responses.** These responses are critically damped and typical of full-state feedback designs.

Figure 3.20, good results were achieved with no further optimization. Figures 3.22 and 3.23 show these results.

3.5.3. Summary of Design Results

A number of controllers have been presented to show the iterative nature of the design methodology and to show an application of several methods to meeting the requirements. Selecting designs for flight test implementation required a review of these results. Figure 3.24 shows a comparison of all the designs. The controllers shown in boldface in Figure 3.24 were selected as candidates for flight. The three designs with integral-error control show a comparison of three sizes of compensators (full-order, reduced order and output feedback). The 3^{rd} order controller without integral control was included to see if the decoupling matrix

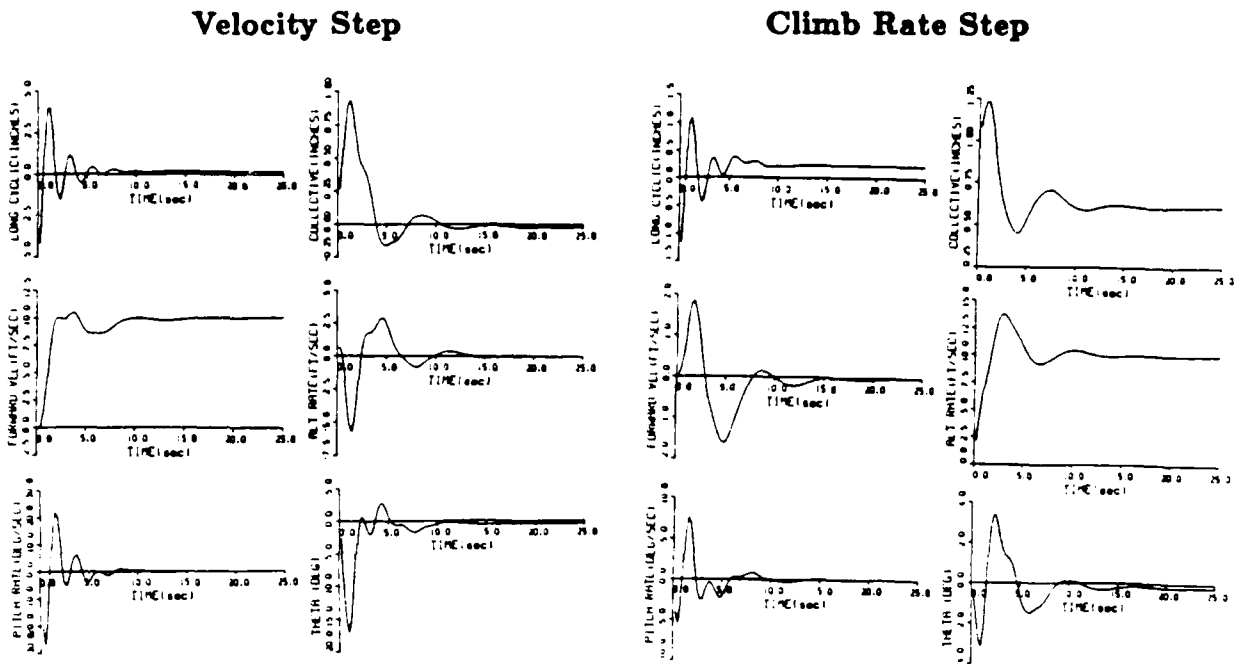


Figure 3.19: **Longitudinal Full Order Integral Controller Time Responses.** This design was also selected for flight evaluation.

$$D = \begin{bmatrix} -5.74 & -12.51 & .041 & .103 \\ -1.23 & -10.44 & -.057 & .013 \end{bmatrix} \tag{3.36}$$

System	Eigenvalues	Damping
Open Loop(F)	$-40.0, -40.0, -2.54, .503, -.105 \pm j.276$	$1, 1, 1, unstable, .35$
Closed Loop$(F - G\,D\,H_m)$	$-39.4, -37.52, -2.07 \pm j.213, -.72, -.47$	$1, 1, .99, 1, 1$

Figure 3.20: **Longitudinal Output Feedback Controller.** Since the measurements (q, θ, $\dot{h}$, and u) are almost the same as the states of a 4^{th} order aircraft model (u, w, q, and $theta$), the output gains were set to the full-state feedback gains from a 4^{th} order regulator design.

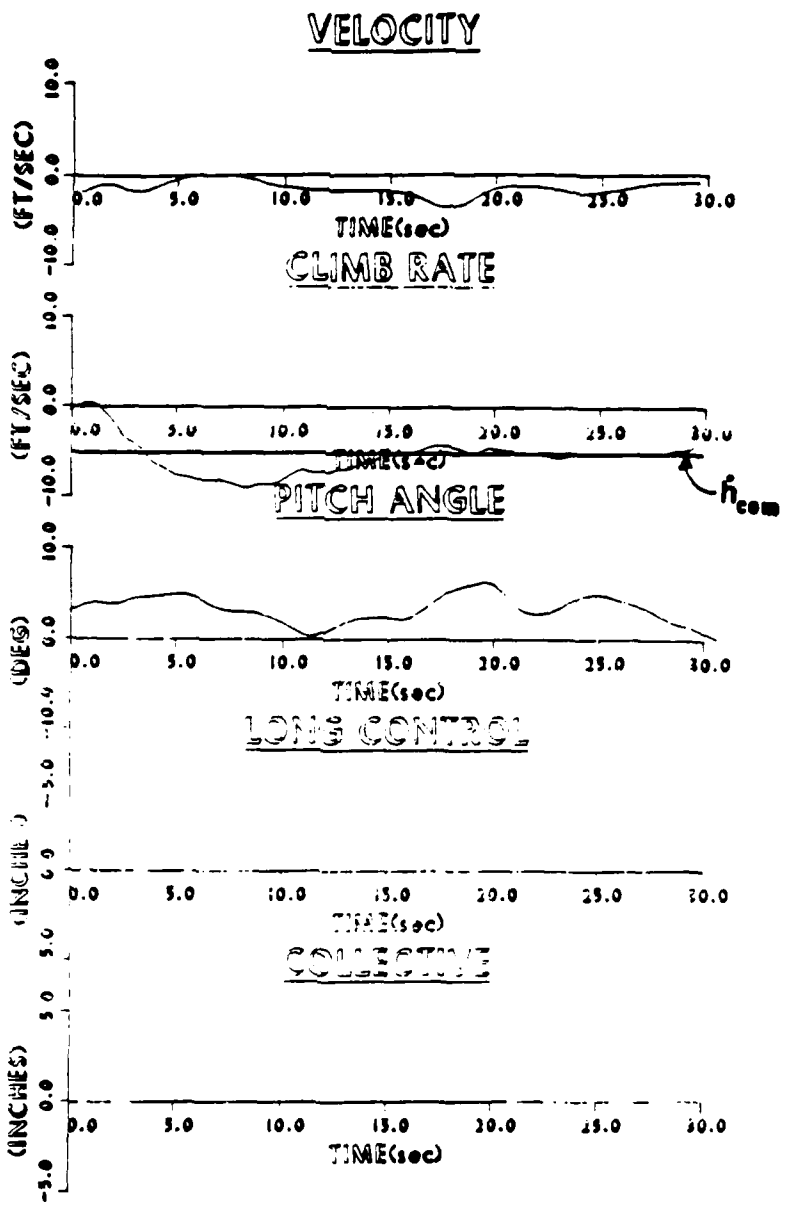


Figure 3.32: **Output Feedback Compensator Flight Response to Climb Rate Command.** As in Figure 3.29, the overall loop gains to the actuators were decreased by half from the analytical design.

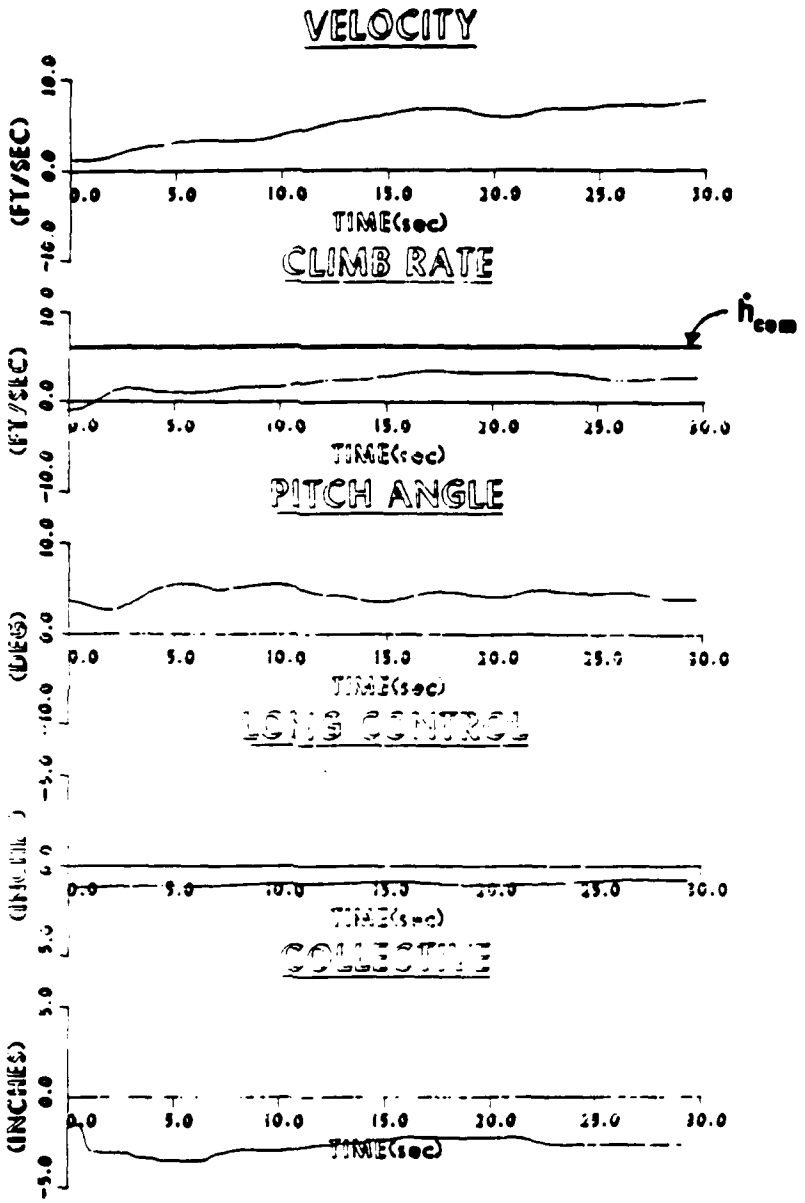


Figure 3.33: **Third Order Flight Response to Climb Rate Command without Integral Control.** This figure documents the effectiveness of the integral control loops. It also emphasizes the need for better models of the CH-47.

Chapter 4.

Hover Controller for the CH-47

4.1. Design Goals and Constraints

With the experience gained in the design of the longitudinal CAS, a more difficult task was selected for the next flight experiment. The second application of the methodology was the design of a translational velocity command, precision hover hold control system for the CH-47. This system was to provide the pilot with "split-axis" control of translational velocity in an heading-oriented inertial coordinate frame. "Split-axis" here means the pilot could select either a velocity command or position-hold control mode in each of the three translational degrees of freedom of the aircraft. As in Chapter 3, this design required decoupling of the three axes of interest. Figure 4.1 shows the coordinate system used for the design. As in Chapter 3, this control law is somewhat unconventional since the pilot's workload is reduced by removing him from inner loop attitude control tasks. For this system, forward velocity is controlled by longitudinal stick displacement, side velocity by lateral stick, and vertical velocity by the collective lever. If a particular pilot control is within a small distance of the neutral position (the detent), then the system enters a position-hold mode for that axis. Yaw rate control, using the

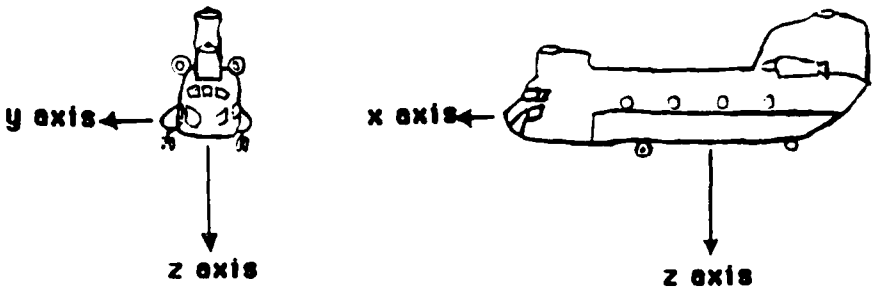


Figure 4.1: **Hover Control System Coordinate System.** The x, y, and z coordinates are in a heading-oriented inertial system.

standard CH-47 SAS, comes from the pedals. No operational helicopter control systems have this type of capability although the concept was tried on a modified CH-47 helicopter during early work on the Army's Heavy Lift Helicopter(HLH) concept.[13] This type of control law has numerous advantages with potential applications such as:

- Search and rescue
- Shipboard operations
- Slung load operations

Before proceeding with the description of how the design methodology was applied, the design constraints must be mentioned. As in the longitudinal CAS design, the TR-48 was used to filter the aircraft motion sensor data. The inertial velocity and position measurements came from either a laser or radar tracker on the ground. This was necessary since the onboard inertial navigation system(INS) had a drift of several knots ($\approx 5 ft/sec$) and provided only position data (no velocities). To avoid major changes in the flight software, a general form of the controller was programmed in the flight computer which provided for a flexible control structure.

$$\begin{bmatrix} \dot{u}\,(ft/sec) \\ \dot{v}\,(ft/sec) \\ \dot{w}\,(ft/sec) \\ \dot{p}\,(radians/sec) \\ \dot{q}\,(radians/sec) \\ \dot{r}\,(radians/sec) \\ \dot{\theta}\,(radians) \\ \dot{\phi}\,(radians) \\ \dot{\delta}_{e_{actual}}\,(inches) \\ \dot{\delta}_{c_{actual}}\,(inches) \\ \dot{\delta}_{a_{actual}}\,(inches) \end{bmatrix} = F \begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \\ \theta \\ \phi \\ \delta_{e_{actual}} \\ \delta_{c_{actual}} \\ \delta_{a_{actual}} \end{bmatrix} + G \begin{bmatrix} \delta_{e_{desired}}\,(inches\ of\ longitudinal\ stick) \\ \delta_{c_{desired}}\,(inches\ of\ collective\ lever) \\ \delta_{a_{desired}}\,(inches\ of\ lateral\ stick) \end{bmatrix}$$

$$y_m(measurements) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ p \\ q \\ r \\ \theta \\ \phi \end{bmatrix} \tag{4.1}$$

Figure 4.2: **CH-47 Hover Model.** This is the conventional 8^{th} order model described in Appendix G.

4.2. Linear Model and Basic Control Structure

The 8^{th} order basic airframe model is described in Appendix G. This model was augmented with three actuator states as shown in Equation 3.2. These actuator states allow the the designer to penalize control rate, a necessary step to avoid nuisance disengages of the research control system during flight. Figure 4.2 shows this model as it was used for the control law synthesis. One item to note is that there is no yaw control in the model. The yaw SAS was approximated by adding -1 to the N_{δ_r} element of the "F" matrix of Figure 4.2 and Appendix G.

Since the longitudinal CAS showed the necessity of integral-error control, the hover control structure had to accommodate this capability. Figure 4.3 shows the hover controller. Unlike the longitudinal CAS, integral control decoupling is not done in the inner loop. Instead, ***PID*** (Proportional Integral Derivative) outer loops are closed separately to $\dot{x}_{command}$, $\dot{y}_{command}$, and $\dot{z}_{command}$, which act as controls for these outer loops. The velocity command system is the inner loop. Setting up the control structure in this way had several advantages:

- The inner loop and outer loop designs, both fairly complicated, could be separated.
- By using the desired outputs as controls, the magnitude of the outer loop gains became physically meaningful. This proved to be important later when these gains were adjusted during flight to achieve good performance.
- Keeping the inner loop separate made the mode switching between pilot velocity command and ***PID*** control easier.

Before the full-order design began, the system was scaled in equivalent units.

4.3. Model Scaling

The hover model was scaled into the same units as used in the longitudinal CAS design. *Ft/sec* remained *ft/sec*; *radians* and *radian/sec* became .01 *radians* and .01 *radians/sec*; and *inches* of control became .1 *inches* of control. Figures 4.4 and 4.5 show the model before and after scaling. The ROPTSYS computer program did this scaling.

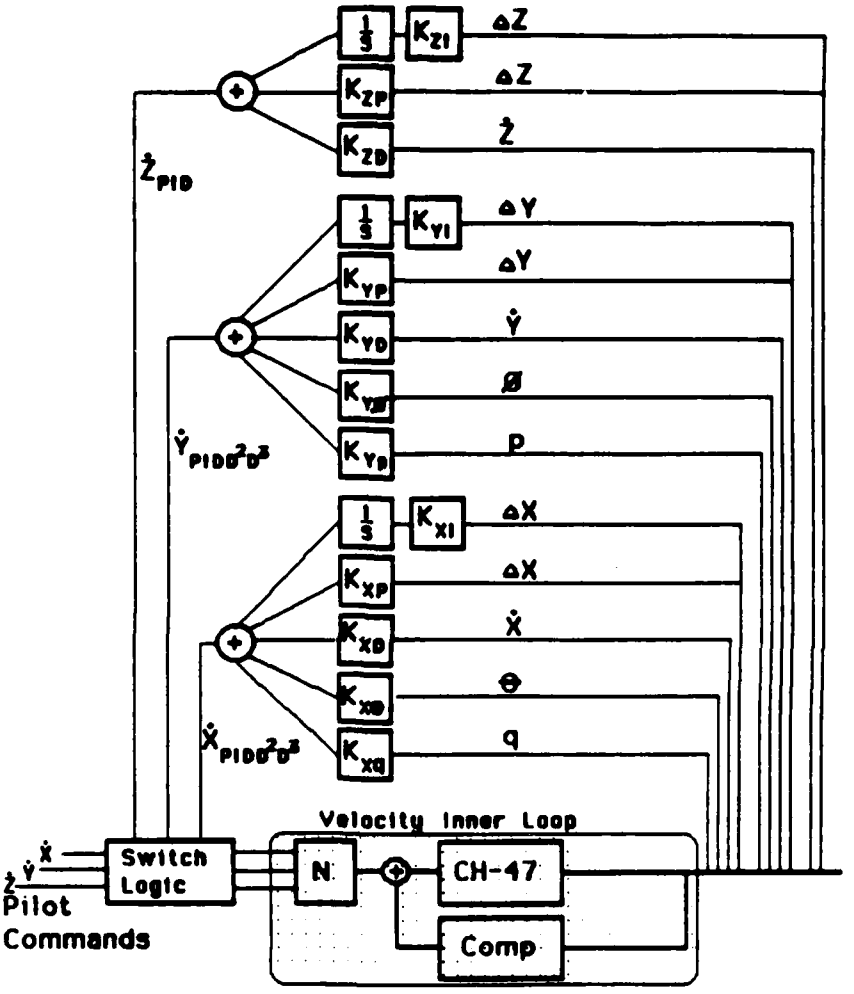


Figure 4.3: **Hover Control System.** The control system is divided into a distinct inner loop (velocity command) and outer loops which provide position hold using *PID* control.

State Equations

$$\dot{x} = F\,x + G\,u$$

$$F = \begin{bmatrix}
-0.021 & -0.00085 & 0.0326 & 0.0205 & 2.585 & -0.106 & -31.986 & 0.0 & 0.114 & 0.939 & 0.0 \\
-0.00019 & -0.137 & 0.00265 & -1.494 & 0.00414 & -0.165 & 0.0292 & 31.99 & 0.0118 & 0.0635 & 1.159 \\
0.0248 & 0.00374 & -0.296 & 0.0419 & 0.435 & 0.362 & -3.71 & 0.0 & 0.303 & -8.062 & .00002 \\
-0.00013 & -0.00652 & 0.00058 & -0.716 & 0.0382 & -0.0708 & 0.0 & 0.0 & -0.00596 & -0.0142 & .432 \\
0.00925 & 0.00017 & 0.00234 & 0.0427 & -1.23 & -0.00433 & 0.0 & 0.0 & 0.329 & 0.019 & 0.0 \\
0.00039 & -0.00112 & 0.00027 & -0.0544 & -0.158 & -1.0 & 0.0 & 0.0 & 0.0461 & -0.00037 & .0425 \\
0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.00788 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 1.0 & 0.00091 & 0.116 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -40.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -40.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -40.0
\end{bmatrix}$$

$$G = \begin{bmatrix}
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
40.0 & 0.0 & 0.0 \\
0.0 & 40.0 & 0.0 \\
0.0 & 0.0 & 40.0
\end{bmatrix}$$

$$Q = \begin{bmatrix}
24.6 & 0 & 0 \\
0 & 24.6 & 0 \\
0 & 0 & 9.98
\end{bmatrix}$$

$$R = \begin{bmatrix}
2.6 \times 10^{-2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2.6 \times 10^{-2} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2.6 \times 10^{-2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1.293 \times 10^{-4} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 3.232 \times 10^{-5} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 8.079 \times 10^{-6} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 3.232 \times 10^{-7} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.232 \times 10^{-7}
\end{bmatrix}$$

Figure 4.4: **CH-47 Hover Linear Model.** The last 3 states are 40 radian actuator models.

State Equations

$$\dot{x}_{scaled} = F_{scaled} x_{scaled} + G_{scaled} u_{scaled}$$

$$F_{scaled} = \begin{bmatrix}
-0.021 & -0.00085 & 0.0326 & 0.000205 & .02585 & -0.00106 & -.31986 & 0.0 & 0.0114 & 0.0939 & 0.0 \\
-0.00019 & -0.137 & 0.00265 & -.1494 & 0.0000414 & -0.00165 & 0.000292 & .3199 & 0.00118 & 0.00635 & .1159 \\
0.0248 & 0.00374 & -0.296 & 0.000419 & 0.00435 & 0.00362 & -.0371 & 0.0 & 0.0303 & -.8062 & .000002 \\
-0.013 & -0.652 & 0.058 & -0.716 & 0.0382 & -0.0708 & 0.0 & 0.0 & -0.0596 & -0.142 & 4.32 \\
0.925 & 0.017 & 0.234 & 0.0427 & -1.23 & -0.00433 & 0.0 & 0.0 & 3.29 & 0.19 & 0.0 \\
0.039 & -0.112 & 0.027 & -0.0544 & -0.158 & -1.0 & 0.0 & 0.0 & 0.461 & -0.0037 & .425 \\
0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.00788 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 1.0 & 0.00091 & 0.116 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -40.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -40.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -40.0
\end{bmatrix}$$

$$G_{scaled} = \begin{bmatrix}
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 \\
40.0 & 0.0 & 0.0 \\
0.0 & 40.0 & 0.0 \\
0.0 & 0.0 & 40.0
\end{bmatrix}$$

$$Q_{scaled} = \begin{bmatrix}
24.6 & 0 & 0 \\
0 & 24.6 & 0 \\
0 & 0 & 9.98
\end{bmatrix}$$

$$R_{scaled} = \begin{bmatrix}
2.6 \times 10^{-2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2.6 \times 10^{-2} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2.6 \times 10^{-2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1.293 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 3.232 \times 10^{-1} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 8.079 \times 10^{-2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 3.232 \times 10^{-2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.232 \times 10^{-2}
\end{bmatrix}$$

Figure 4.5: **CH-47 Scaled Hover Linear Model.** After scaling, the weak coupling terms in the F matrix become very obvious.

4.4. LQG Design and Compensator Order Reduction

The scaled 11^{th} order model shown in Figure 4.5 was used by the ROPTSYS program to calculate the full-order compensator. At this point, an important simplification should be emphasized. The system was to control $\dot{x}$, $\dot{y}$, and $\dot{z}$ which are inertial velocities. The model as used included u, v, and w which are body axis airmass velocities. The implicit assumption, needed to facilitate the design, was that the two sets of velocities were equal:

$$\begin{aligned} \dot{x} &= u \\ \dot{y} &= v \\ \dot{z} &= w \end{aligned} \tag{4.2}$$

This assumption is reasonable only if θ and ϕ remain small, which they must for safe hover in a large helicopter such as the CH-47. The outputs, y_o, weighted in the performance index:

$$P.I. = \int_0^\infty (y_o^T A y_o + u^T B u)\, dt \tag{4.3}$$

were the three velocities (u, v, and w) and the three control rates. Since the system had been scaled, the weighting matrices, A and B, were just 6×6 and 3×3 identity matrices. Using these assumptions and criterion, the resulting full-order compensator is shown in Figure 4.6.

Based on these data, especially the modal cost M_2, the 11^{th} order compensator was reduced to 5^{th} order. This represents a significant reduction of complexity in the resulting compensator. The 11^{th} order design had 121 independent gains while the 5^{th} order compensator had only 55. After the order reduction, the compensator

$$F_{min} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -80.615 & -15.033 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -55.126 & -11.587 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -5.78 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2.74 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2.33 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.48 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.01 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -.203 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -.104 \end{bmatrix}$$

$$K_{min}(unscaled) = \begin{bmatrix} .0032 & -.00096 & .0024 & -.0112 & -.63 & .023 & -32.4 & .097 \\ .0294 & .0183 & -.00274 & .21 & -17.3 & -2.76 & 59.8 & 4.3 \\ -.002 & -.06 & .0127 & -.442 & .166 & -.99 & .587 & -61.4 \\ .0072 & -.32 & -.023 & -3.6 & -1.64 & -10.62 & -17.3 & 170.1 \\ -.077 & .0115 & .616 & -.074 & -.079 & -.56 & -1.45 & -6.74 \\ -.028 & .56 & .17 & 5.56 & 4.24 & 16.2 & 1.34 & -100.8 \\ -.054 & -.0134 & .136 & -.233 & 25.4 & 3.68 & 3.78 & .33 \\ .087 & -.0084 & -.69 & .146 & .00055 & .81 & .50 & .053 \\ -.0003 & -.02 & .0033 & -.18 & .83 & -.34 & -.48 & -2.19 \\ .000054 & -.034 & -.0022 & -.21 & -.023 & -.56 & .018 & 5.16 \\ -.014 & -.00012 & -.00117 & -.00145 & .125 & .0133 & 4.37 & -.028 \end{bmatrix}$$

$$C_{min}(unscaled) = \begin{bmatrix} 0.0 & -.1 & -.0128 & -.00328 & .0226 & -.00358 & -.1 & .00119 & -.024 & .00067 & -.1 \\ .0011 & -.0036 & .0089 & .00294 & -.1 & .0044 & -.0041 & -.1 & -.012 & .012 & -.0108 \\ .065 & .0056 & 0.0 & -.1 & .0148 & -.1 & .0186 & -.0058 & -.1 & -.1 & -.00247 \end{bmatrix}$$

System Eigenvalues										
	Open Loop(F)		Regulator(F-GC)		Estimator(F-KH)		Compensator(F-GC-KH)		Compensator Measures	
Mode	Real	Imag	Real	Imag	Real	Imag	Real	Imag	M_1	M_2
1	-1.41	0	-.406	.085	-7.63	4.85	-7.52	4.91	4.47	.59
2	-1.07	0	-.406	-.085	-7.63	-4.85	-7.52	-4.91	-	-
3	-.902	0	-.382	.072	-6.04	4.6	-5.79	4.64	5.58	.96
4	-.297	0	-.382	-.072	-6.04	-4.6	-5.79	4.64	-	-
5	.079	.46	-.839	.4	-5.88	0	-5.78	0	.65	.11
6	.079	-.46	-.839	-.4	-.975	0	-2.73	0	1.18	.43
7	.062	.46	-1.2	.33	-.11	0	-2.32	0	.3	.13
8	.062	-.46	-1.2	-.33	-.12	0	-1.48	0	.69	.47
9	-40.0	0	-1.3	.11	-40.0	0	-1.01	0	.032	.032
10	-40.0	0	-1.3	-.11	-40.0	0	-.20	0	.063	.31
11	-40.0	0	-.98	0	-40.0	0	-.10	0	.046	.45

Figure 4.6: **Hover Full Order Compensator Design Results.** The open loop has two unstable modes which are divergent pitch and roll oscillations. Figure 4.7 is the reduced and reoptimized compensator based on this full order design. Order reduction was based on the M_2 terms shown here.

was optimized using the RSANDY program. Figure 4.7 shows the optimal reduced order compensator including the closed loop roots. This optimization step was difficult since the initial guess (the original 11^{th} order compensator reduced to 5^{th}) was quite unstable. This caused numerical overflows on the VAX computer used to run the RSANDY program. Convergence to a stable 5^{th} order compensator was finally achieved after numerous iterations of the outputs and the output weightings of Equation 4.3. Figure 4.7 shows these outputs, y_o, and the elements of the diagonal A and B matrices. This figure also shows another aspect of the difficulty of this optimization. The entire C matrix was allowed to vary which meant that there were 60 gains being adjusted by the RSANDY program, 5 more than the 55 independent gains of a minimal realizaton.

The simulation step responses are shown in Figures 4.9, 4.10, and 4.11. All three velocity responses look good but the pitch angle damping has several overshoots. A modal analysis later confirmed this by showing the mode at $-.50 \pm j2.03$ of Figure 4.7 to be strongest in θ. With the velocity inner loop regulator designed, the feedforward matrix was calculated for direct command of $\dot{x}$, $\dot{y}$, and $\dot{z}$. The PID outer loop gains could now be designed.

4.5. Outer Loop Design

As discussed in Section 4.2, the velocity command inner loop and the PID outer loop were separate. With the inner loop velocity controller set, the outer loop design began. This approach (inner loop first then outer loop) is common in the development of operational aircraft autopilots except that the inner loops are normally designed classically using incremental loop closure. Initially, these PID

$$F_{min} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -17.93 & -4.90 & 0 & 0 & 0 \\ 0 & 0 & -1.98 & 0 & 0 \\ 0 & 0 & 0 & -.815 & 0 \\ 0 & 0 & 0 & 0 & -6.31 \end{bmatrix}$$

$$K_{min} = \begin{bmatrix} -.035 & -.050 & -.040 & -.56 & 1.96 & .216 & 3.22 & -74.7 \\ -.037 & -.74 & .071 & -.28 & -1.6 & -7.63 & -19.7 & 169.0 \\ -.58 & .140 & -.23 & .19 & 28.9 & 16.05 & 1.35 & -100.3 \\ -.45 & .0079 & 1.45 & .068 & 6.5 & 1.20 & 2.64 & 7.16 \\ .018 & -.0058 & -.003 & -.029 & -.36 & -.66 & 13.7 & 8.32 \end{bmatrix}$$

$$C_{min} = \begin{bmatrix} -.092 & .0080 & -.13 & -.020 & -5.14 \\ .026 & -.0045 & -.029 & .022 & -.52 \\ .44 & -.021 & -.062 & .00078 & -.42 \end{bmatrix}$$

$$N = \begin{bmatrix} -.071 & -.0061 & .013 \\ .0070 & .0021 & -.079 \\ -.012 & .059 & -.0034 \end{bmatrix}$$

Closed Loop Eigenvalues				
Real Part	Imag Part	Damping	Freq(rad/sec)	Freq(Hz)
-39.99	.0085	1.0	39.99	6.37
-39.99	-.0085	1.0	39.99	6.37
-40.01	0.0	1.0	40.01	6.37
-7.14	0.0	1.0	7.14	1.14
-2.37	2.77	.65	3.65	.58
-2.37	-2.77	.65	3.65	.58
-.50	2.03	.24	2.09	.33
-.50	-2.03	.24	2.09	.33
-.54	1.59	.32	1.68	.27
-.54	-1.59	.32	1.68	.27
-.55	.50	.74	.74	.12
-.55	-.50	.74	.74	.12
-.61	.43	.82	.75	.12
-.61	-.43	.82	.75	.12
-.90	0.0	1.0	.90	.14
-.22	0.0	1.0	.22	.0035

Performance Index Data		
Outputs or Controls	Units	Weighting
u	ft/sec	5×10^3
v	ft/sec	5×10^3
w	ft/sec	5×10^3
p	rad/sec	1×10^5
q	rad/sec	1×10^5
r	rad/sec	1×10^5
$\dot{\delta}_e$	$inches/sec$	1×10^5
$\dot{\delta}_c$	$inches/sec$	1×10^5
$\dot{\delta}_a$	$inches/sec$	1×10^5
$\dot{u}$	ft/sec^2	5×10^3
$\dot{v}$	ft/sec^2	5×10^3
$\dot{w}$	ft/sec^2	5×10^3
δ_e	$inches$	1×10^3
δ_c	$inches$	1×10^3
δ_a	$inches$	1×10^3

Figure 4.7: **Reduced Order Hover Compensator.** The poorly damped modes ($s = -.5 \pm j2.03$) dominate the pitch angle response as Figure 4.9 shows.

outer loop gains were set using the RSANDY program. This approach did not work well as the convergence was very slow. The alternative was to set the gains intuitively. This approach was actually quite reasonable if one recalls the control structure of Figure 4.3. The gains were set by determining how much velocity would be reasonable to use to correct a given position error. For instance, if the aircraft were 10 feet from the desired hover point and a pilot would be willing to use 5 ft/sec of $\dot{y}$ then $K_{\dot{y}\delta_y}$ would be .5 $\frac{ft/sec}{ft\ error}$. This approach worked well for the y and z axes but failed for the x axis. For this axis, a conventional transfer function analysis was used to set the 5 outer loop gains which were then adjusted in simulation.

4.5.1. Transfer Function Analysis for X Axis Outer Loop

The first step in the process required a transfer function from $u_{command}$ to u_{actual} for the helicopter with velocity inner controller. This came from use of Bernard's code for calculating the matrix of transfer functions for any MIMO linear dynamic system.[7] These transfer functions were 11^{th} order so to make the process tractable, the NAVFIT program at NASA Ames simplified them to 3^{rd} order. The following transfer function resulted:

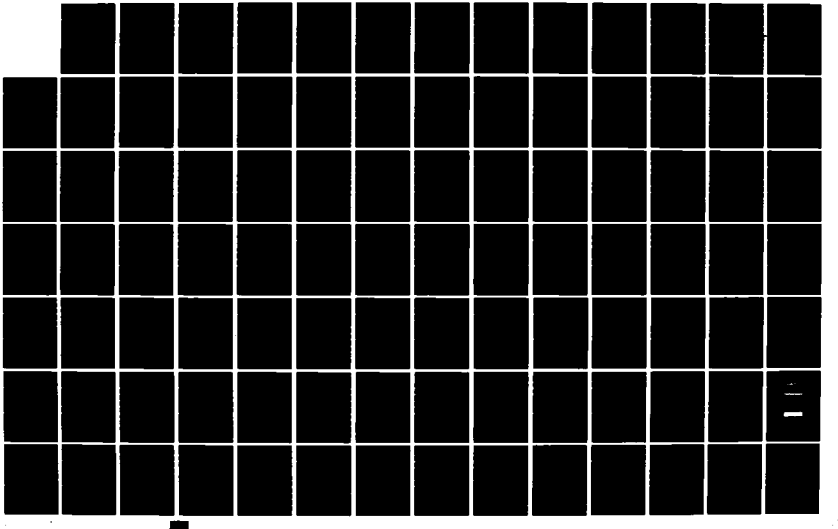
$$\frac{u_{actual}}{u_{command}} = \frac{.94}{(s+.24)\left[(s+.5)^2+1.87^2\right]} \tag{4.4}$$

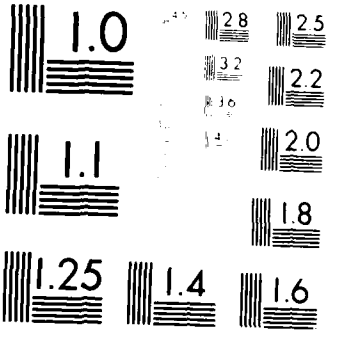
To justify the use of θ and q as second and third derivatives of x, consider the longitudinal equation of motion from Etkin: [14]

$$F_x - mg\ sin\theta = m\left[\dot{u} + (q_B^E + q)\ w - (r_B^E + r)\ v\right] \tag{4.5}$$

where if the following assumptions are made:

AD-A151 946 A MODERN CONTROL DESIGN METHODOLOGY WITH APPLICATION TO 2/3
THE CH-47 HELICOPTER(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH R D HOLDRIDGE JAN 85
UNCLASSIFIED AFIT/CI/NR-85-33D F/G 1/4 NL





$q_B^E \approx 0$ Earth rotation negligible

$r_B^E \approx 0$ Earth rotation negligible

$qw \approx 0$ Second order effect

$rv \approx 0$ Second order effect

$sin\ \theta \approx \theta$ Small angle assumption

$F_x \approx 0$ Reasonable for a helicopter in hover

$\dot{u} \approx \ddot{x}$ Small angle assumption(θ)

then these simplications result:

$$\begin{aligned} \ddot{x} &\approx -g\theta \\ \ddot{u} &\approx -g\dot{\theta} \approx -gq \end{aligned} \tag{4.6}$$

The actual gain setting is done by including the following for the $PIDD^2D^3$ controller in Figure 4.8.

$$PIDD^2D^3 = K_P x + K_I \frac{x}{s} + K_D x s + K_{D^2} x s^2 + K_{D^3} x s^3 \tag{4.7}$$

where

$$\begin{aligned} K_P &= K_{i\delta_x} \\ K_I &= K_{i\int x} \\ K_D &= K_{i\dot{x}} \\ K_{D^2} &= K_{i\theta} \\ K_{D^3} &= K_{iq} \end{aligned} \tag{4.8}$$

Rewriting Equation 4.7 as:

$$PIDD^2D^3 = K_{D^3}\left(s^4 + \frac{K_{D^2}}{K_{D^3}}s^3 + \frac{K_D}{K_{D^3}}s^2 + \frac{K_P}{K_{D^3}}s + \frac{K_I}{K_{D^3}}\right)\frac{1}{s} \tag{4.9}$$

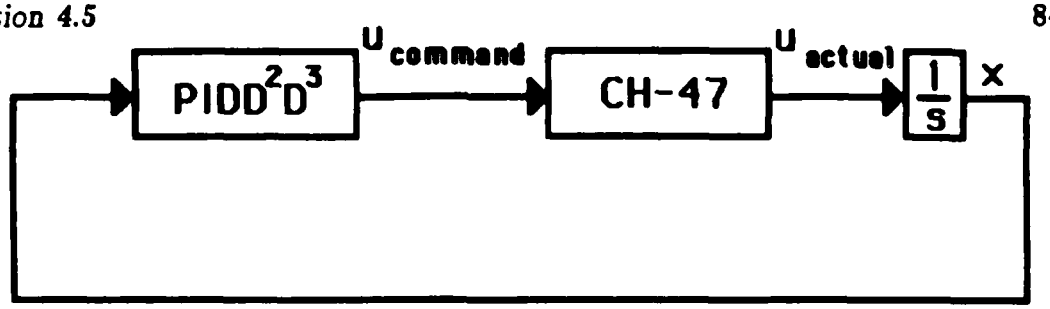


Figure 4.8: **Transfer Function Analysis.** The $PIDD^2D^3$ compensation was calculated to cancel the two lightly damped poles of Equation 4.4 and move the two poles at the origin to the left.

then the four numerator zeros of the $PIDD^2D^3$ controller were selected to cancel the lightly damped poles of Equation 4.4 and to draw the two poles at the origin to the left. The gain $K_D{}^3$ was selected for good speed of response. Following are the gains calculated:

$$\begin{aligned} K_P = K_{XP} &= -1.085 \frac{fps}{(ft\ error)} \\ K_I = K_{XI} &= -.131 \frac{fps}{(ft\ sec\ error)} \\ K_D = K_{XD} &= -2.94 \frac{fps}{fps} \\ K_{D^2} = K_{X\theta} &= .551 \frac{fps}{deg} \\ K_{D^3} = K_{XQ} &= .393 \frac{fps}{(deg/sec)} \end{aligned} \tag{4.10}$$

4.5.2. Outer Loop Simulation

With these gains as a starting point, the time responses were improved using the onboard simulation in the flight computer. Although this approach ("tweeking" the gains) may seem somewhat unscientific, it was appropriate for several reasons:

- Extensive use of the onboard simulation had the added advantage of helping discover many errors in the flight software before actual flight.
- Pilot comments concerning the performance and response characteristics could be better incorporated into the design.
- Working directly with the fixed point digital computer avoided the additional time and effort required to digitize and scale the continuous design.
- The transient-free switching between the velocity command and position hold modes could be developed. This is discussed in the next section.

The PID gains coming from this simulation are shown below:

$$\begin{array}{ccc} K_{XP} = -.75\ \frac{fps}{(ft\ error)} & K_{YP} = -1.0\ \frac{fps}{(ft\ error)} & K_{ZP} = -2.0\ \frac{fps}{(ft\ error)} \\ K_{XI} = -3.8 \times 10^{-4}\ \frac{fps}{(ft\ sec\ error)} & K_{YI} = -7.6 \times 10^{-4}\ \frac{fps}{(ft\ sec\ error)} & K_{ZI} = -1.9 \times 10^{-3}\ \frac{fps}{(ft\ sec\ error)} \\ K_{XD} = -3.0\ \frac{fps}{fps} & K_{YD} = -.18\ \frac{fps}{fps} & K_{ZD} = -4.5\ \frac{fps}{fps} \\ K_{X\theta} = .2\ \frac{fps}{deg} & K_{Y\phi} = 0.0\ \frac{fps}{deg} & \\ K_{XQ} = 2.0\ \frac{fps}{(deg/sec)} & K_{YP} = 0.0\ \frac{fps}{(deg/sec)} & \end{array} \tag{4.11}$$

Time responses from simulation are shown in Figures 4.9 to 4.13. These figures show the results of both the inner and outer loop designs.

4.6. Flight Test Implementation

The flight implementation of the hover control system was based on the software and flight procedures developed for the longitudinal CAS flight test. As before, the sensor data were filtered by the TR-48 analog computer before being digitized and sent to the Sperry digital computer. There were two areas where the hover controller was quite different from the longitudinal system and required new flight capabilities. The first was the inertial position and velocity data and the second was the transient-free switching required to make the transition from

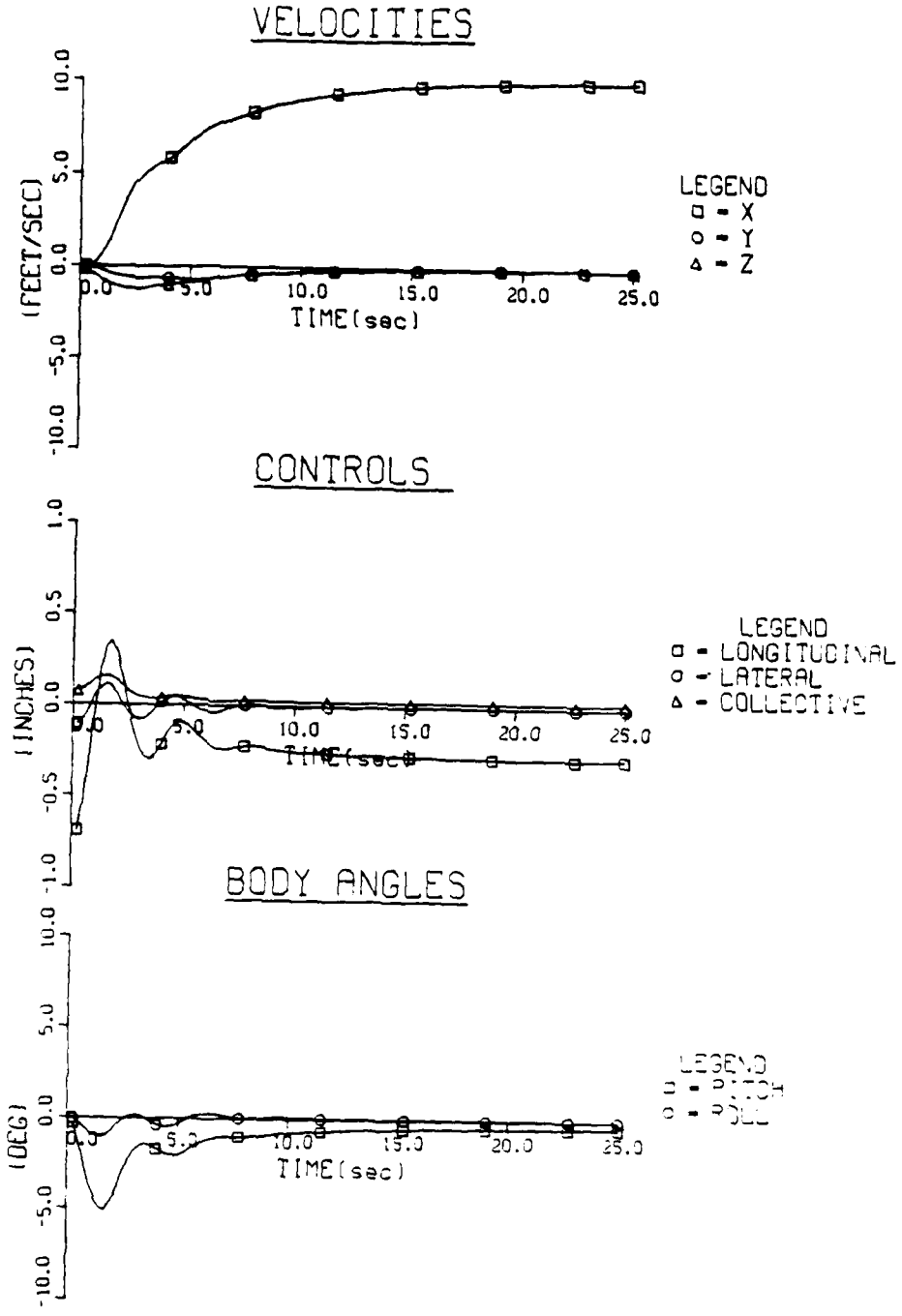


Figure 4.9: **Hover Forward Velocity Step Command in Simulation.** The poor damping is evident in the pitch angle response.

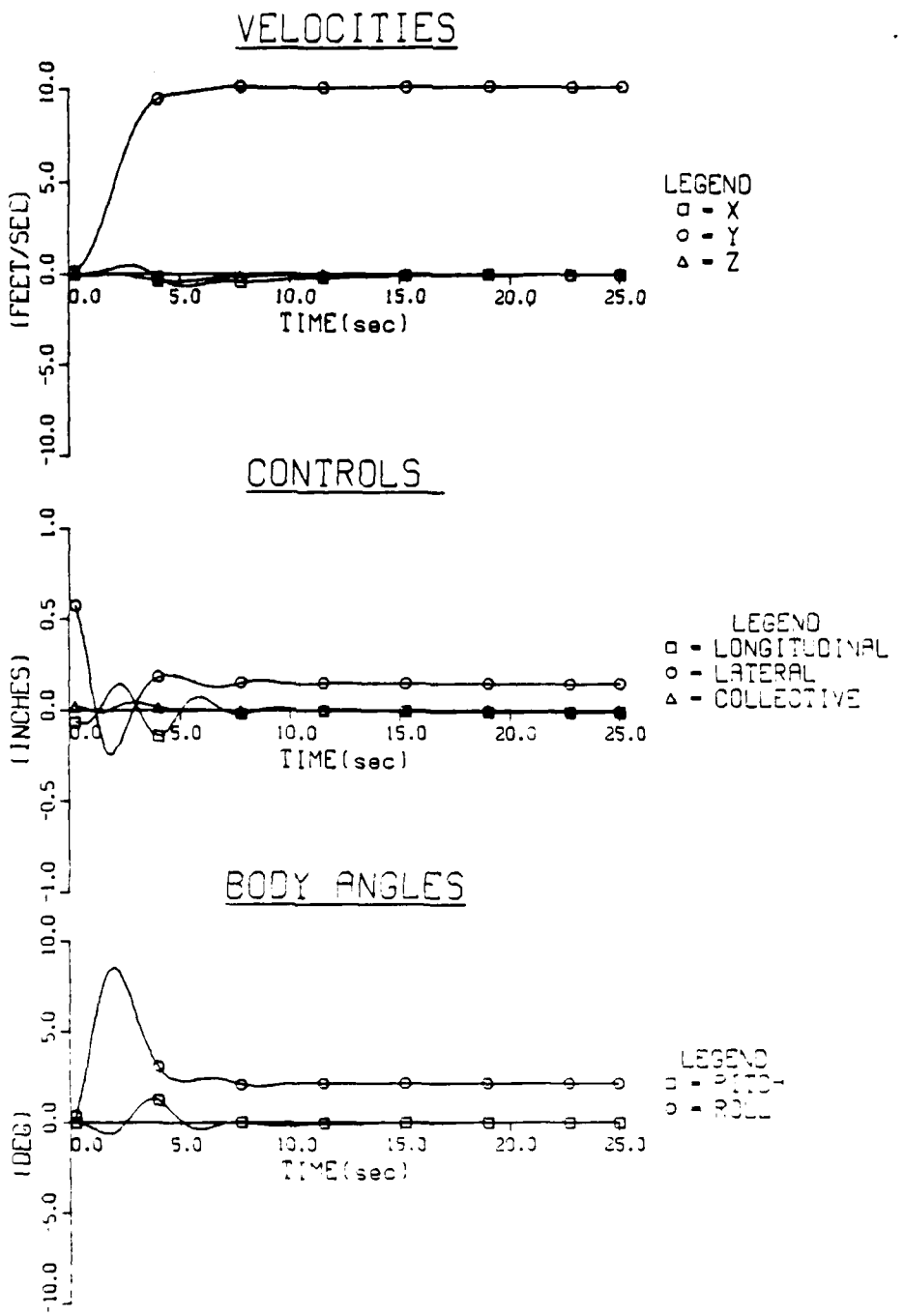


Figure 4.10: **Hover Side Velocity Step Command in Simulation.** Side velocity performance is adequate but the poorly damped pitch mode is also excited.

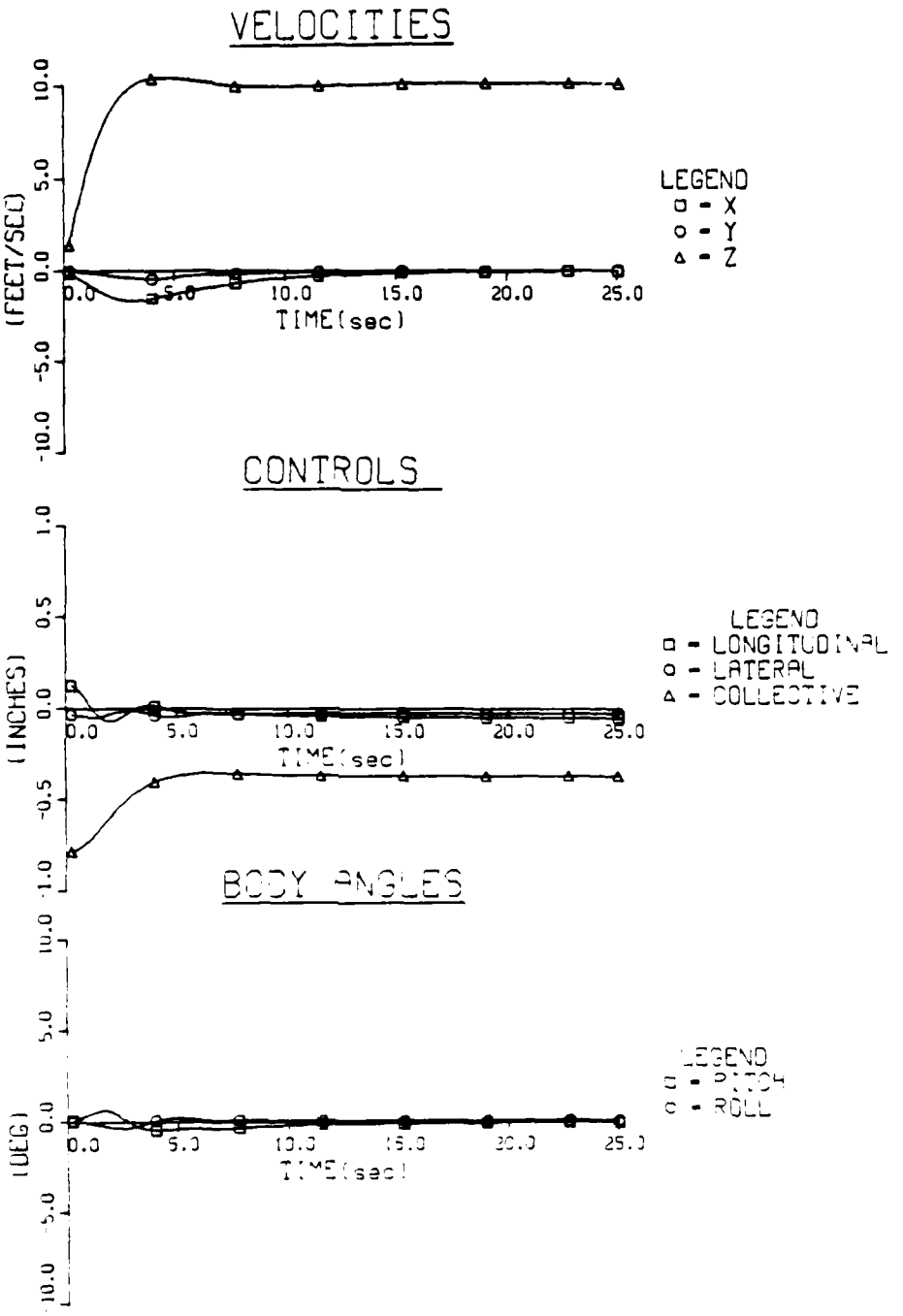


Figure 4.11: Hover Vertical Velocity Step Command in Simulation. The heave response is adequate and does not excite the pitch modes as strongly as the lateral velocity response of Figure 4.10

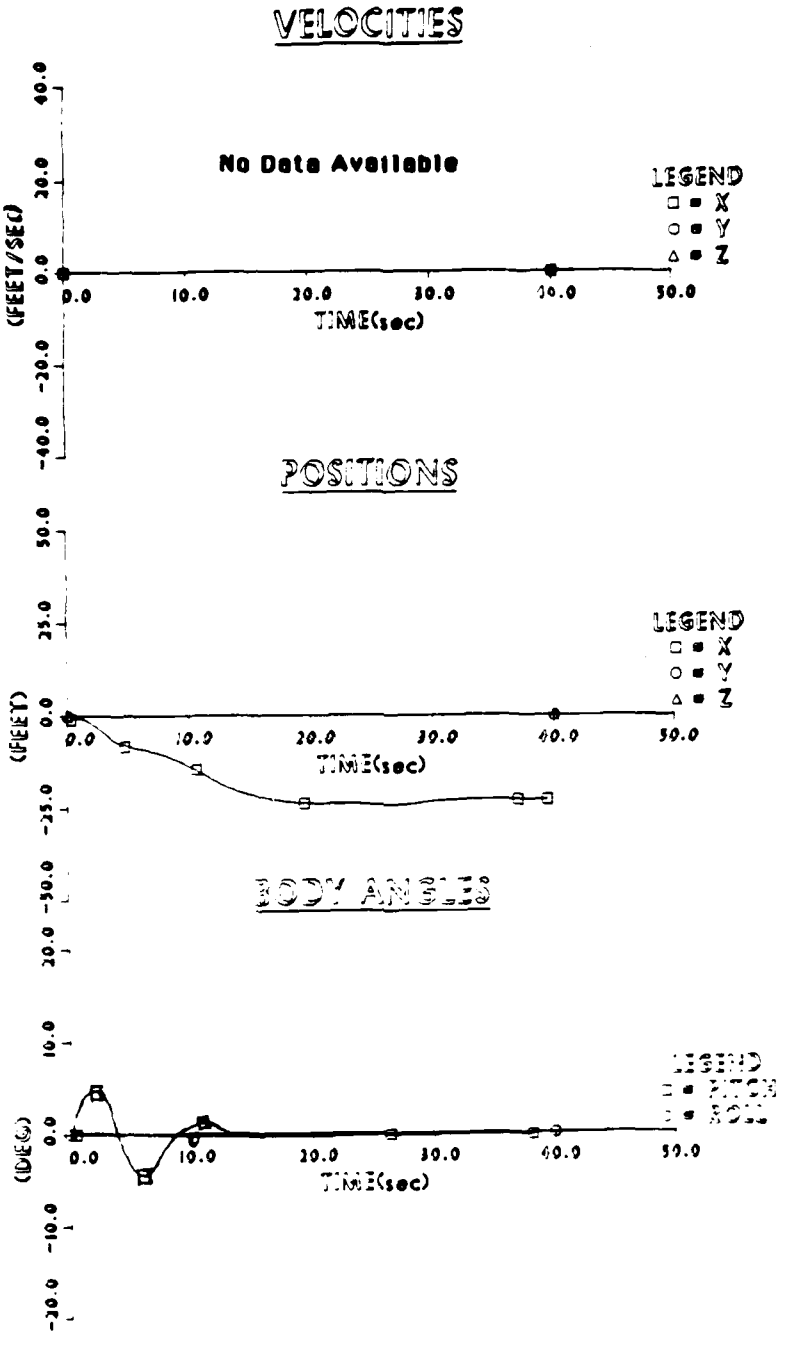


Figure 4.12: **Hover Forward Position Step Command in Simulation.** This response came from the onboard simulation and shows the the outer loop performance using the flight software.

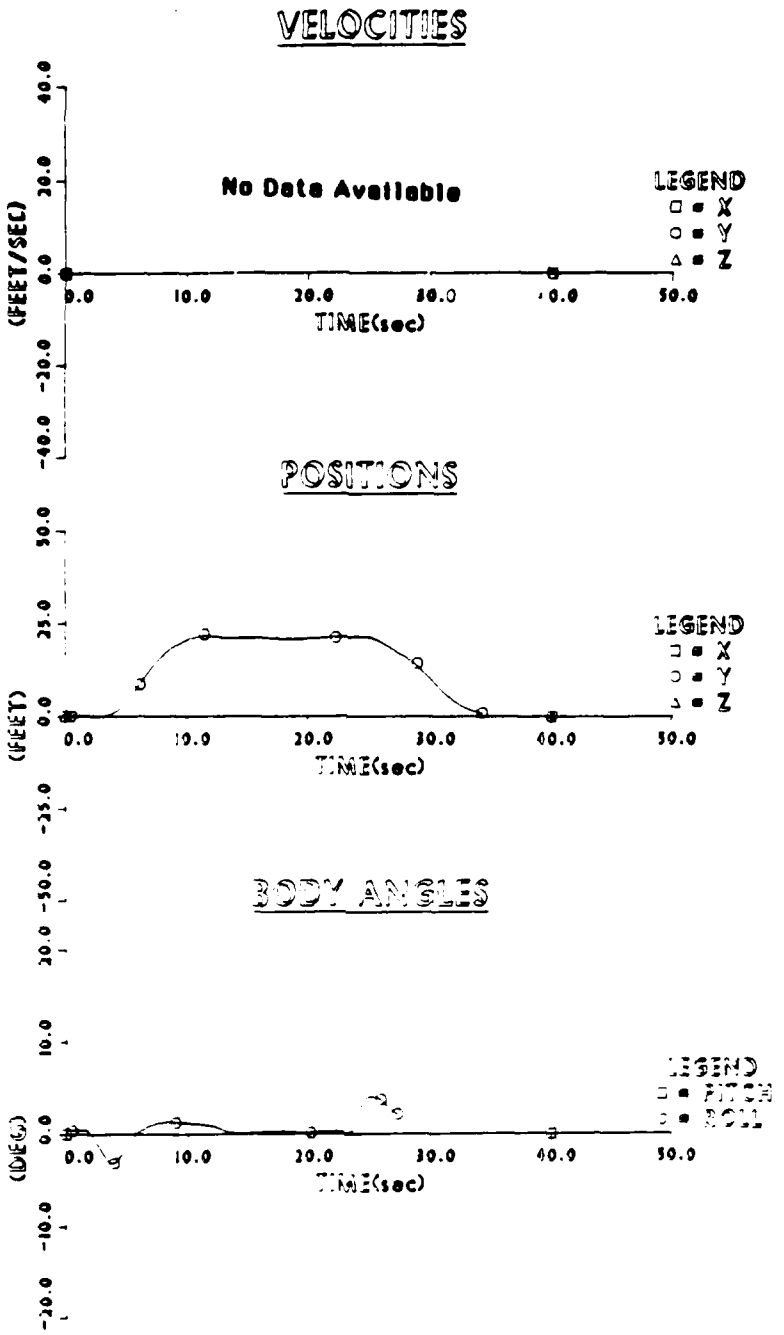


Figure 4.13: **Hover Lateral Position Step Command in Simulation.** Again, the flight software shows good outer loop performance.

pilot velocity command to automatic position hold.

4.6.1. Inertial Velocity and Position Data

Although the hover controller uses the inertial data (x, $\dot{x}$, y, $\dot{y}$, z, and $\dot{z}$) as it does the other measurements, a considerable effort was required to get these data.[1] Since the INS positions drifted so quickly and there were no inertial velocities available from the INS, an alternative source for these data was needed. The ground based tracker at Crows Landing was able to provide these data using a ground-to-air telemetry link that was specially developed for this program. The steps required to make these ground based position measurements usable by the control laws are described below:

- The laser or radar tracker measured position of the aircraft in a runway based polar coordinate system.
- These measurements of azimuth angle (Az), elevation angle (El), and range (r) were telemetered from the ground tracking station to the helicopter.
- These data, as well as tracker status information, were decoded and scaled into units common to the rest of the flight software.
- The Az, El, and range data were converted to a runway based rectangular coordinate system with a new origin located over the runway.
- These data were used with aircraft accelerometer data (rotated into the runway coordinate frame) in a second order complementary filter to estimate x, $\dot{x}$, y, $\dot{y}$, z, and $\dot{z}$. Figure 4.14 shows the block diagram of this complementary filter.

[1] This work was done primarily by Bill Hindson of NASA Ames

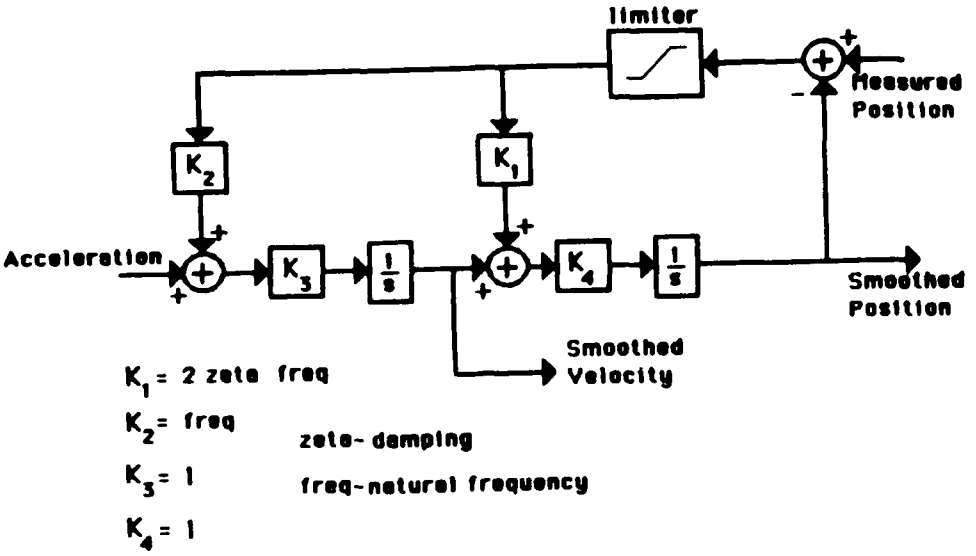


Figure 4.14: **Second Order Complementary Filter.** The filter uses acceleration and position to determine the smoothed position and velocity needed by the control law.

- These smoothed values of x, $\dot{x}$, y, $\dot{y}$, z, and $\dot{z}$ were then rotated through the aircraft heading angle to the heading-oriented inertial frame required by the control system.

- Based on the tracker status information coming from the ground and based on data reasonableness checks, an algorithm kept the inertial data consistent during short term tracker breaklocks. For longer term breaklocks, the experimental control system was disengaged to avoid the large control motions caused by trying to follow bad data. Initially, the laser tacker was used since it provided more precise range information (1 - 2 foot acurracy). Unfortunately, the laser had frequent and unpredictable breaklocks which made the data essentially unusable in the control loop. Because of this inability to hold lock, the radar tracker was used for the flight test although its accuracy was only 5 - 10 feet.

4.6.2. Transient-Free Switching

Since this control system had both manual and automatic capability in the three body axes, a way was needed to transition smoothly among these different control modes. This task was complicated by the following characteristics of the control system:

- The pilot had the freedom to change heading at any time.
- The x coordinate of the desired hover point had to follow the x coordinate of the actual position when the pilot was commanding $\dot{x}$ velocity. At the same time, the helicopter had to hold both y and z position in the heading inertial frame.
- Same as above in the y and z directions.
- A detent on the pilot controls was needed. If the pilot's control was less than the detent value, that axis was in position hold mode; else, the pilot was commanding a velocity.

Figures 4.15, 4.16, and 4.17 show the switching logic for the three axes. The assembly code, shown in Appendix I, implements this logic.

4.7. Flight Test Results

The hover flight testing was done at the Crows Landing test facility. The testing was limited to this location since the system required the use of the radar tracker at Crows. Unlike the longitudinal CAS control system, the hover controller was very difficult to debug and make operational. The flight testing was divided into three phases to accommodate these difficulties. The first phase developed the

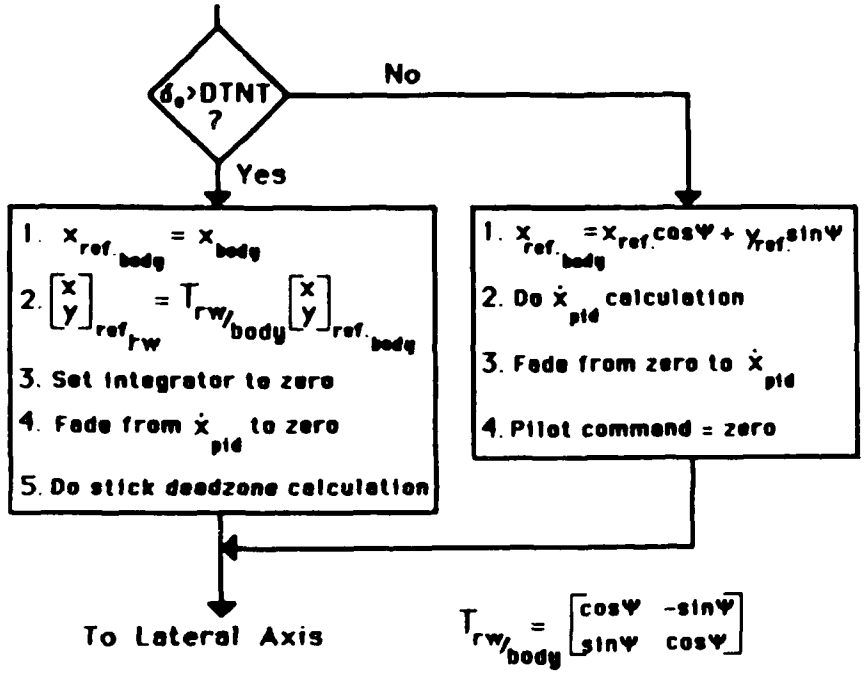


Figure 4.15: **X Axis Transient-Free Switching Logic.** The transformation from body to runway uses heading angle from the INS. The best deadzone or detent value was about .25 inches.

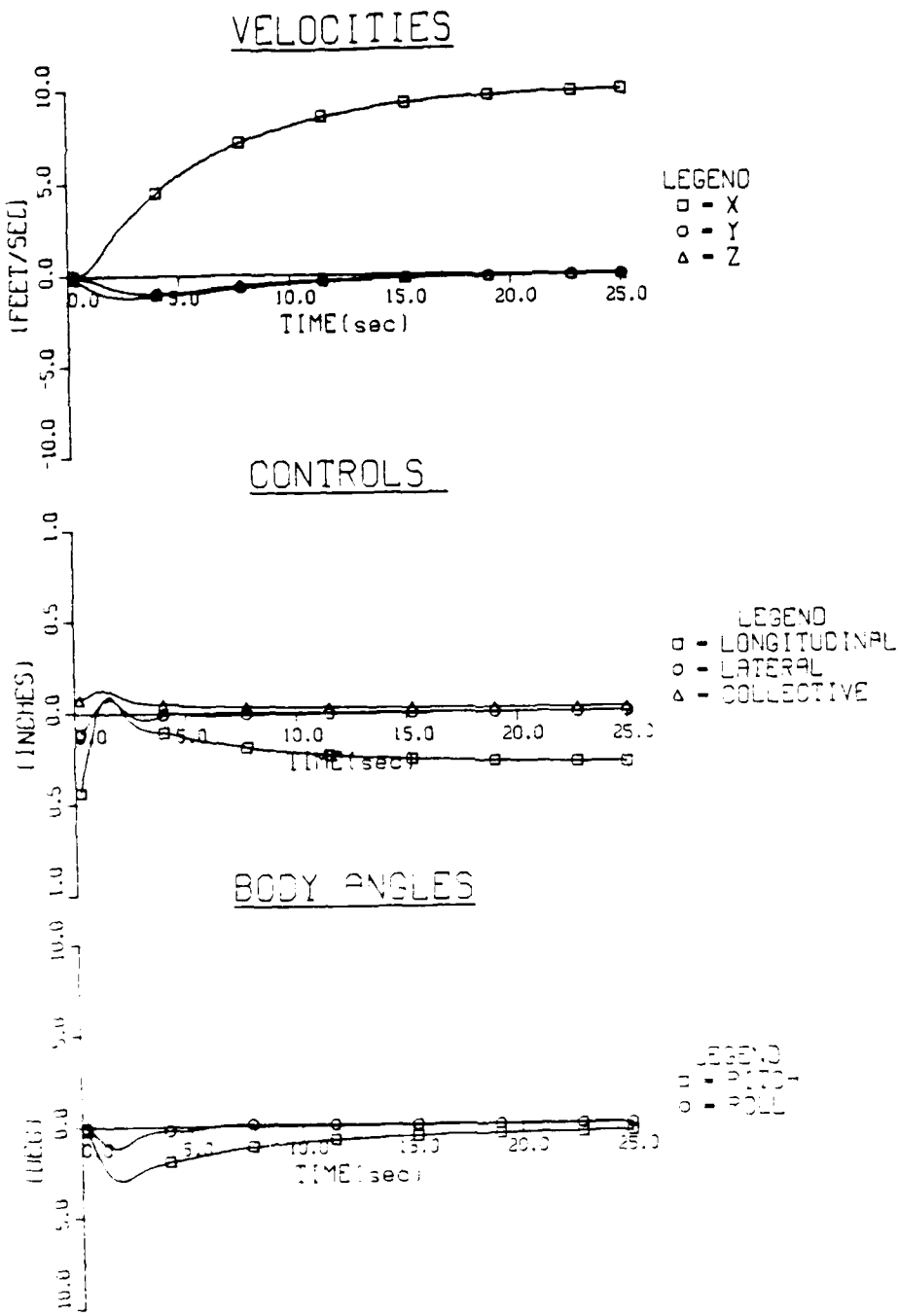


Figure 4.25: **Hover Forward Velocity Step Command in Simulation.** The pitch angle response is improved from the original design of Figure 4.9.

quicken the lateral response while in hold mode hence reducing any coupling due to action in the longitudinal axis. With these two changes to reduce the coupling, and the redesigned inner loop velocity system, the final flight testing began.

4.7.4. Final Closed Loop Flight Test in Hover

With the redesigned controller, the velocity performance was significantly improved. Figure 4.28 shows the response to a command in $\dot{x}$. The poor pitch damping has been eliminated and the coupling to bank angle is gone. The y axis velocity performance remains good as shown in Figure 4.29. The z velocity response remained almost identical to the original design shown in Figure 4.21. The position hold performance is also evident in these figures when the velocity commands are removed and the system reenters the postion hold mode. Figures 4.30 and 4.31 confirm the good hold performance in the y and z axes but the poor damping in position hold in x. The x axis position hold dynamics are dominated by a slow, poorly damped mode ($\zeta \approx .4$ and $\omega \approx 50$ *sec*). A significant amount of flight time was spent adjusting gains in the PID outer loops to improve this x hold performance. Shown below are the final set of outer loop gains which resulted from these efforts. Later flight tests used the integrator in the x axis $PIDD^2D^3$ controller only when the error was less than 40 feet. This improved the damping slightly to about .5.

$$
\begin{array}{ccc}
K_{XP} = -.20 \ \frac{fps}{(ft\ error)} & K_{YP} = -1.0 \ \frac{fps}{(ft\ error)} & K_{ZP} = -2.0 \ \frac{fps}{(ft\ error)} \\
K_{XI} = -7.6 \times 10^{-4} \ \frac{fps}{(ft\ sec\ error)} & K_{YI} = -7.6 \times 10^{-3} \ \frac{fps}{(ft\ sec\ error)} & K_{ZI} = -1.9 \times 10^{-3} \ \frac{fps}{(ft\ sec\ error)} \\
K_{XD} = -2.0 \ \frac{fps}{fps} & K_{YD} = -1.0 \ \frac{fps}{fps} & K_{ZD} = -4.5 \ \frac{fps}{fps} \\
K_{X\theta} = 1.0 \ \frac{fps}{deg} & K_{Y\phi} = -.5 \ \frac{fps}{deg} & \\
K_{XQ} = 3.0 \ \frac{fps}{(deg/sec)} & K_{YP} = -.5 \ \frac{fps}{(deg/sec)} &
\end{array}
\tag{4.12}
$$

$$F_{min} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -17.93 & -4.90 & 0 & 0 & 0 \\ 0 & 0 & -1.98 & 0 & 0 \\ 0 & 0 & 0 & -.815 & 0 \\ 0 & 0 & 0 & 0 & -6.31 \end{bmatrix}$$

$$K_{min} = \begin{bmatrix} -.035 & -.050 & -.040 & -.56 & 2.18 & .216 & 3.23 & -74.7 \\ -.037 & -.74 & .071 & -.28 & -1.57 & -7.63 & -19.7 & 169.0 \\ -.58 & .140 & -.23 & .19 & 29.2 & 16.05 & 1.30 & -100.3 \\ -.45 & .0079 & 1.45 & .068 & 6.52 & 1.20 & 2.57 & 7.16 \\ .018 & -.0058 & -.003 & -.029 & 3.4 & -.66 & 14.03 & 8.32 \end{bmatrix}$$

$$C_{min} = \begin{bmatrix} .07 & .054 & -.0776 & -.0036 & -4.61 \\ .026 & -.0045 & -.029 & .022 & -.52 \\ .44 & -.021 & -.062 & .00078 & -.42 \end{bmatrix}$$

$$N = \begin{bmatrix} -.048 & -.0098 & -.0024 \\ .007 & -.0079 & .0021 \\ 0.0 & -.0034 & .0059 \end{bmatrix}$$

Closed Loop Eigenvalues				
Real Part	Imag Part	Damping	Freq(rad/sec)	Freq(Hz)
-40.2	0.0	1.0	40.2	6.39
-40.01	0.0	1.0	40.01	6.367
-39.99	0.0	1.0	39.99	6.365
-6.007	0.0	1.0	6.007	.956
-2.37	2.70	.66	3.59	.57
-2.37	-2.70	.66	3.59	.57
-.94	2.13	.40	2.33	.37
-.94	-2.13	.40	2.33	.37
-.60	1.51	.37	1.62	.26
-.60	-1.51	.37	1.62	.26
-.55	.48	.75	.73	.116
-.55	-.48	.75	.73	.116
-.62	.402	.84	.74	.117
-.62	-.402	.84	.74	.117
-.90	0.0	1.0	.90	.14
-.17	0.0	1.0	.17	.027

Performance Index Data		
Outputs or Controls	Units	Weighting
u	ft/sec	5×10^3
v	ft/sec	5×10^3
w	ft/sec	5×10^3
p	rad/sec	1×10^5
q	rad/sec	1×10^6
r	rad/sec	1×10^5
$\dot{\delta}_e$	$inches/sec$	1×10^5
$\dot{\delta}_c$	$inches/sec$	1×10^5
$\dot{\delta}_a$	$inches/sec$	1×10^5
$\dot{u}$	ft/sec^2	5×10^3
$\dot{v}$	ft/sec^2	5×10^3
$\dot{w}$	ft/sec^2	5×10^3
δ_e	$inches$	1×10^2
δ_c	$inches$	1×10^2
δ_a	$inches$	1×10^2

Figure 4.24: **Redesigned Hover Compensator.** Only the columns of K associated q and θ and the row of C corresponding to longitudinal control (δ_e) are changed from the initial design of Figure 4.7.

good time responses. In this case, the measurement noise characteristics were left unchanged and an unrealistically high value of the vertical velocity disturbance was used. Specifically, the vertical gust root mean square (rms) was increased from 2.3 ft/sec to 10 ft/sec. Vertical gust was selected since it affects the pitch angle more strongly than the other disturbances. With this one change, the RSANDY program was used to find a new compensator. To speed up the convergence in the RSANDY program, only the columns of the K_{min} matrix associated with measurements of q and θ, and the row of C_{min} associated with the longitudinal control were allowed to vary. This approach was also logical since we wished to keep the vertical and lateral axes unchanged from the first design. The redesigned compensator in shown in Figure 4.24 and can be compared to the initial design in Figure 4.7. Figure 4.25 shows the simulation response of the redesigned velocity command inner loop with the improvement in pitch damping compared to the initial design shown in Figure 4.9. Figures 4.26 and 4.27 show that the y and z responses were essentially unchanged by the redesign.

Since there was nothing in the simulation to suggest that there would be coupling from $\dot{x}$ command to ϕ, the approach to solving this problem was based on the experience gained thus far. Two changes were made to the controller which would have to wait for flight to be evaluated. The first change was the zeroing of the feedforward gain from $\dot{x}$ command to δ_a in the N matrix of Figure 4.3. This was a logical approach to solving the problem since the N matrix was highly dependent on accurate modeling and the longitudinal flight test had already shown the model to be lacking. The other change made to solve this coupling was to include nonzero values for $K_{Y\phi}$ and K_{YP} in the lateral PID outer loop. This change was made to

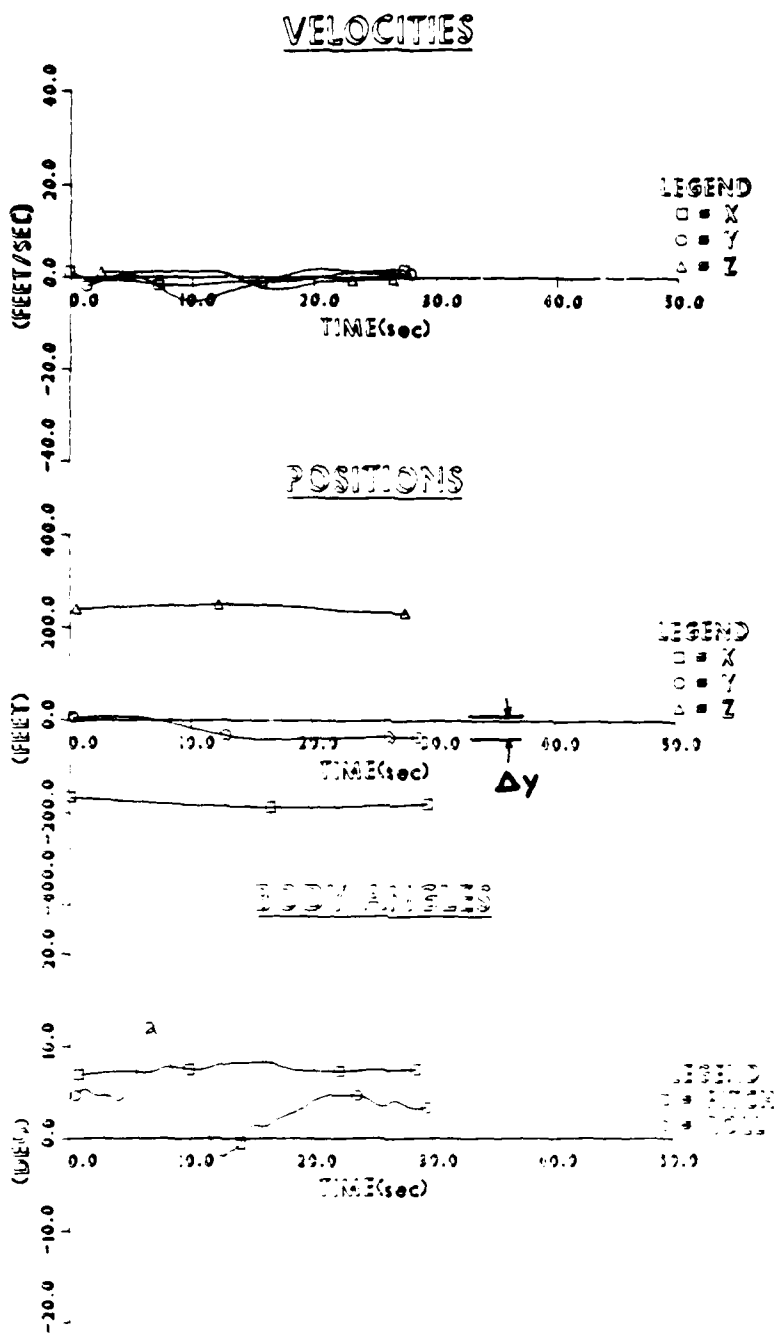


Figure 4.23: **Preliminary Lateral Position Step Command in Flight.** The y position hold performance is well damped and similar to the simulation of Figure 4.13.

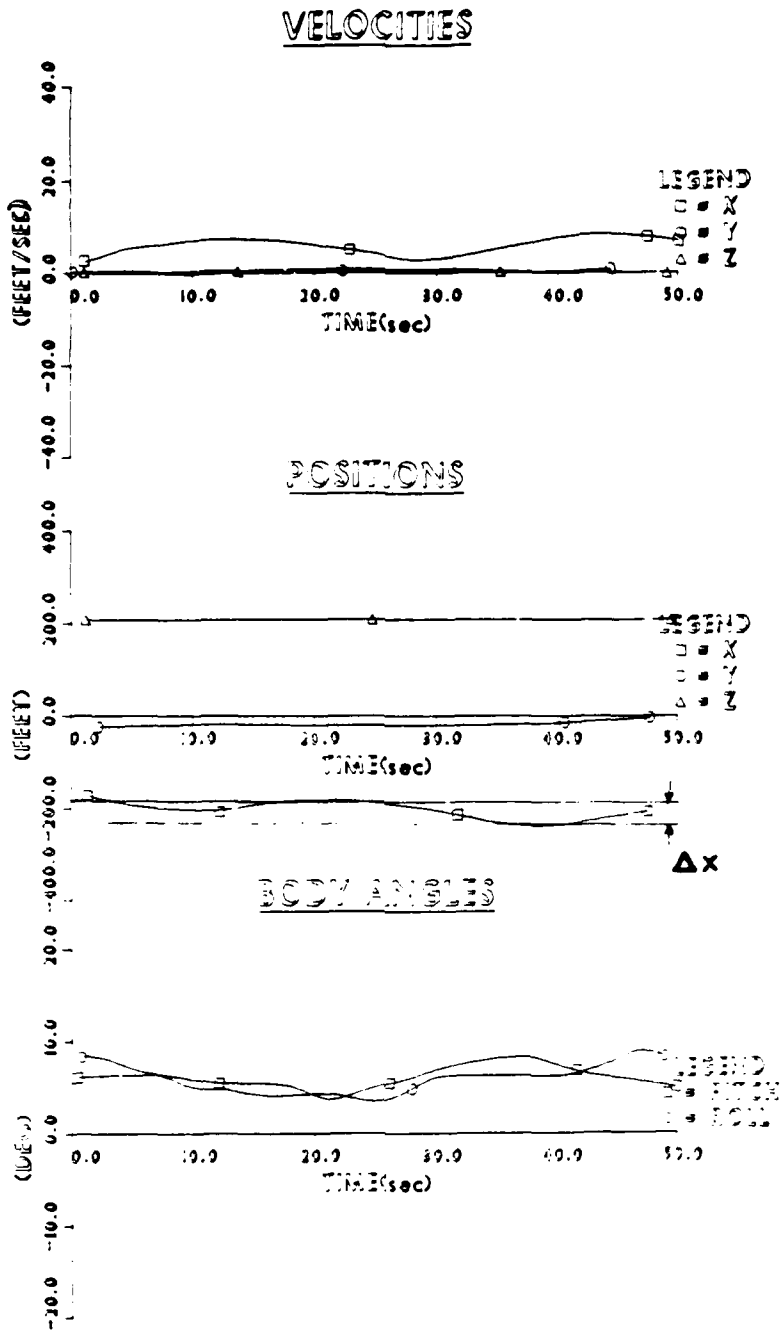


Figure 4.22: **Preliminary Forward Position Step Command in Flight.** The x position damping is very poor ($\zeta \approx .1$), unlike the near critical damping of the simulation shown in Figure 4.12.

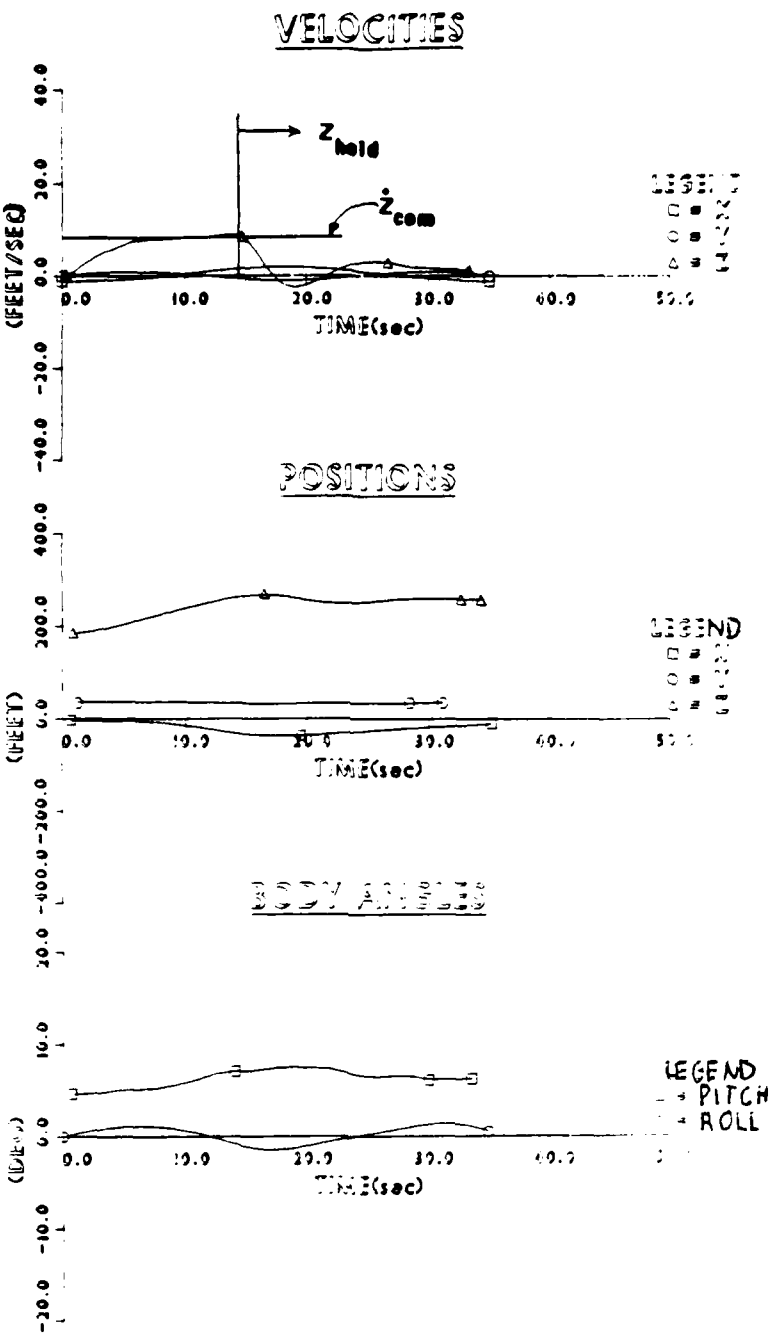


Figure 4.21: **Preliminary $\dot{z}$ Flight Response.** The 5 second time to steady state matches the simulation shown in Figure 4.11.

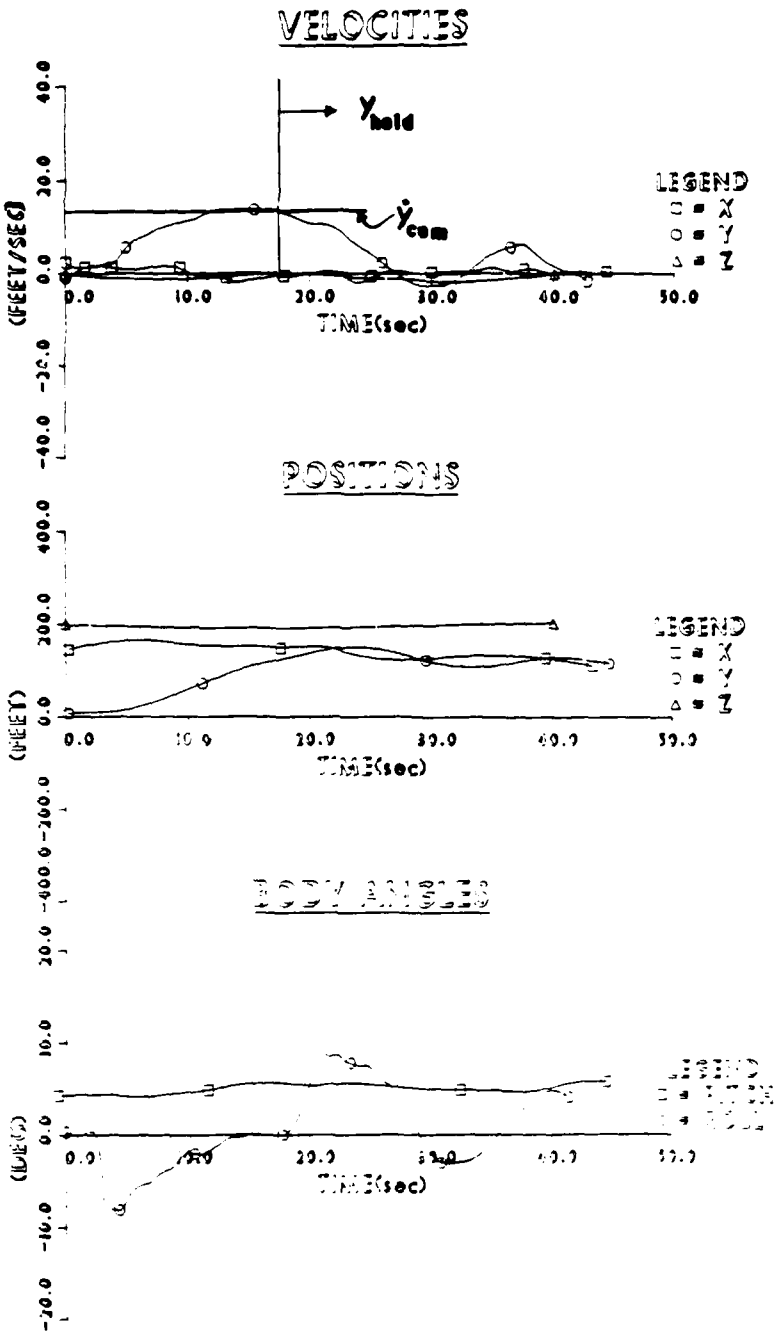


Figure 4.20: **Preliminary $\dot{y}$ Flight Response.** The $\dot{y}$ response is well behaved and similar to the simulation results of Figure 4.10. The peak roll angle is about 8 degrees for both responses to a command of about 10 ft/sec.

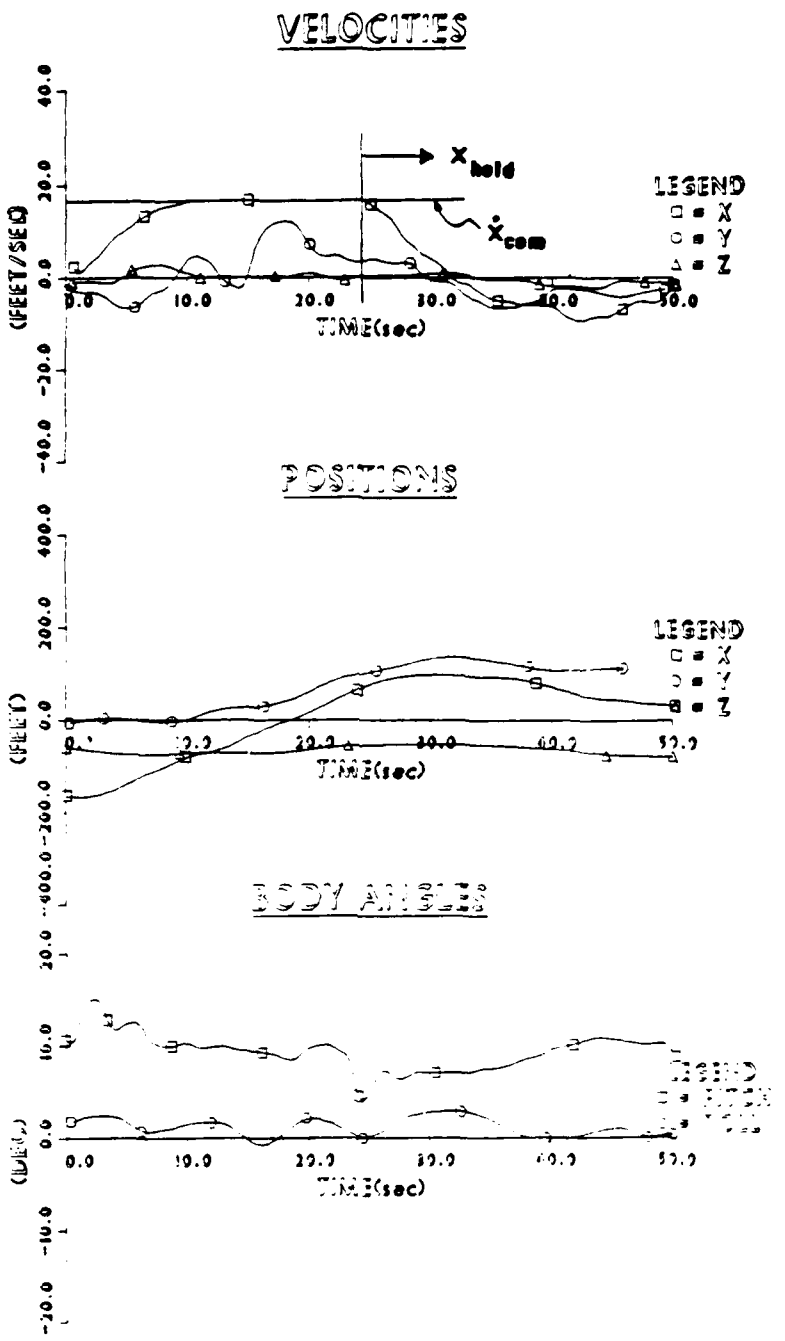


Figure 4.19: **Preliminary $\dot{x}$ Flight Response with Pitch Roll Coupling.** With the instability of Figure 4.18 corrected, the roll coupling, shown here, was discovered.

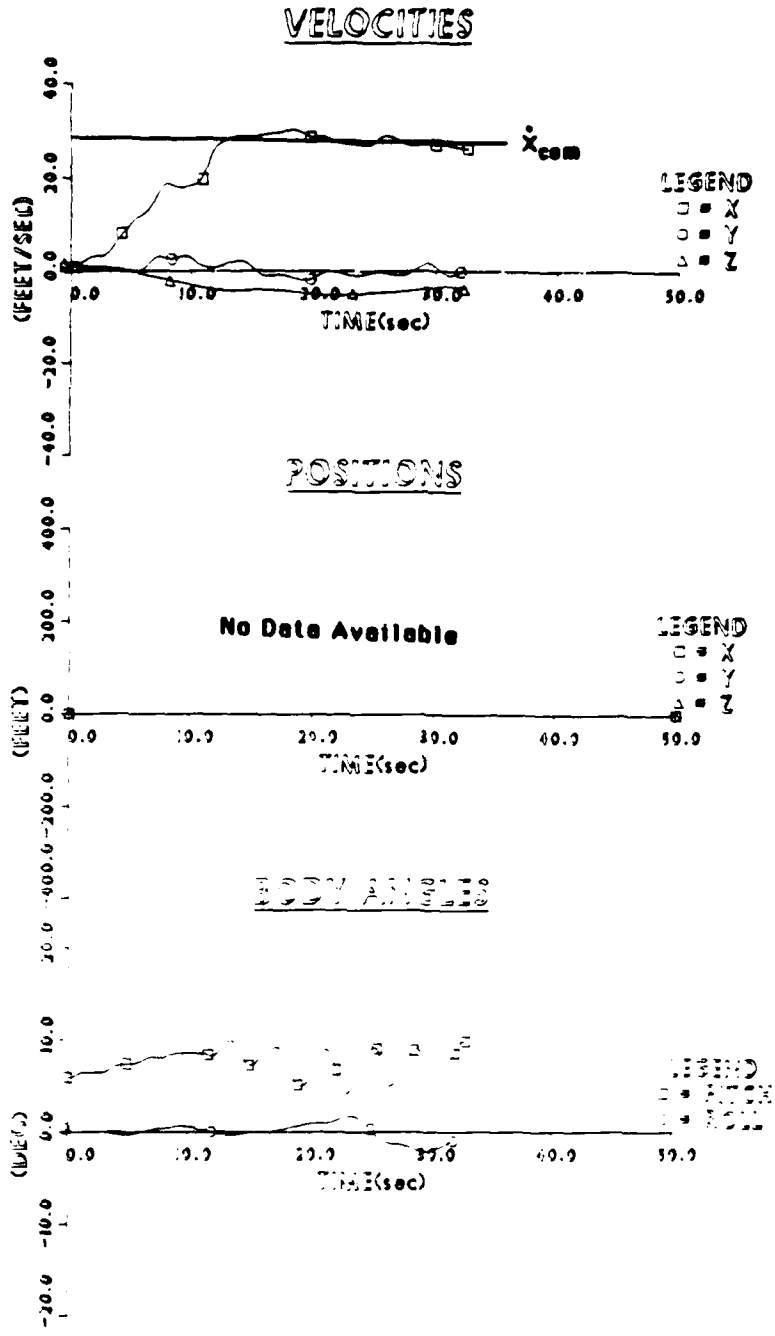


Figure 4.18: **Preliminary $\dot{x}$ Flight Response with Pitch Oscillation.** The poorly damped simulation response of Figure 4.9 became unstable in flight as the pitch angle shows. The decoupling using the outer loops worked well.

system in flight. Figure 4.18 shows the flight results with the extremely poor x axis performance. After a redesign of the velocity inner loop system, described in the next section, another problem was found. There was unacceptable coupling between the x and y axes. The coupling, evident in Figure 4.19, was manifested as roll osscillations resulting from the $\dot{x}$ command. The next section also describes the approach taken to solve this pitch to roll coupling problem. The coupling was one way, however, as seen in Figure 4.20 where the response to a step in $\dot{y}$ is quite acceptable and similar to the simulation results of Figure 4.10. The $\dot{z}$ command capability is also quite good as Figure 4.21 shows. The x position hold performance of Figure 4.22 was very poor due to the low damping ($\approx .1$). Y position (Figure 4.22) was much better damped and faster than x. The z position hold performance was very good with vertical position changes of less than 10 feet during the velocity commands of Figures 4.19, 4.20, and 4.21. Use of the radar tracker data in the inner loops, which was considered risky due to its complexity, worked well throughout the flight test.

4.7.3. Hover Controller Redesign

The redesign of the system was necessitated by bad performance in two modes. First, the x velocity response was slightly unstable in flight. The other problem was the coupling from x velocity command to roll angle. The first problem was handled by redesigning the inner loop velocity control system in order to slow the longitudinal response. This redesign was first attempted by changing the weighting matrices in the RSANDY program to get a better damped longitudinal response. This approach did not work so the technique of Section 3.4.1 was used. Section 3.4.1 described using an arbitrary set of measurement spectral densities to achieve

new capabilities needed by the hover controller including data uplink capability and complementary filtering to get smooth inertial data. The second phase included preliminary flight test which discovered poor velocity performance which necessitated a redesign of the velocity inner loop. The final phase of the flying evaluated the redesigned control system.

4.7.1. Support Systems Development Flying

The complexity of the hover controller required that essentially all the aircraft systems and all the ground support equipment be working in order to exercise the system. A number of flights was required just to ensure that the uplink system and the associated complementary filters were producing good inertial data. Once these systems were operating correctly, the flight testing continued with checks of the mode switching and transient suppression logic while using the real data coming from the complementary filters. It was while doing this work that the laser tracker's poor ability to hold lock was discovered and the decision was make to go with the less accurate radar tracker.

4.7.2. Preliminary Closed Loop Flight Test in Hover

Preliminary closed loop testing included velocity step commands in the three axes and changes in desired position while remaining in the hover hold mode. These closed loop tests confirmed what the longitudinal CAS tests had already shown. The flight responses were less damped than the simulations had predicted. In other words, we couldn't achieve as high a bandwidth in flight as in simulation. This was most evident in the x axis where the well damped velocity response in the simulation (Figure 4.9) turned into a neutrally stable or slightly unstable

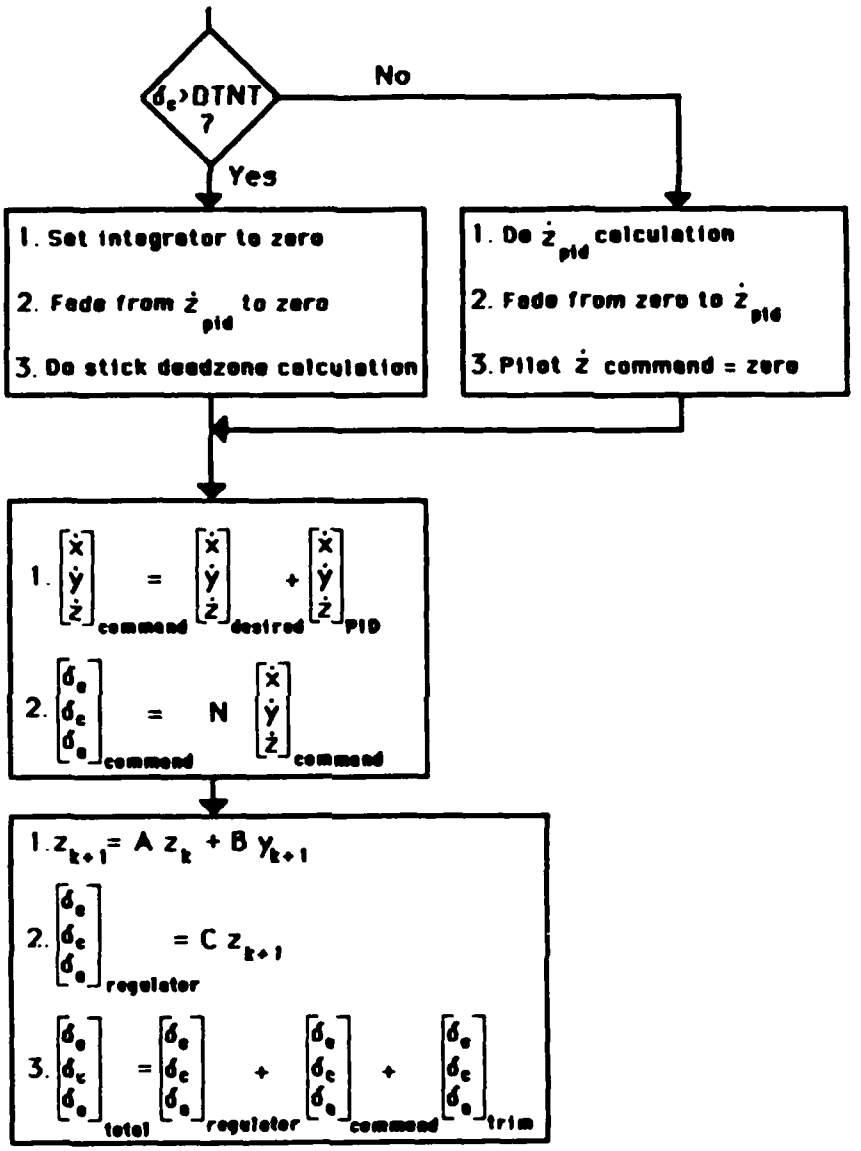


Figure 4.17: **Z Axis Transient-Free Switching Logic.** This figure also shows the rest of the hover controller which was shown in Figure 4.3.

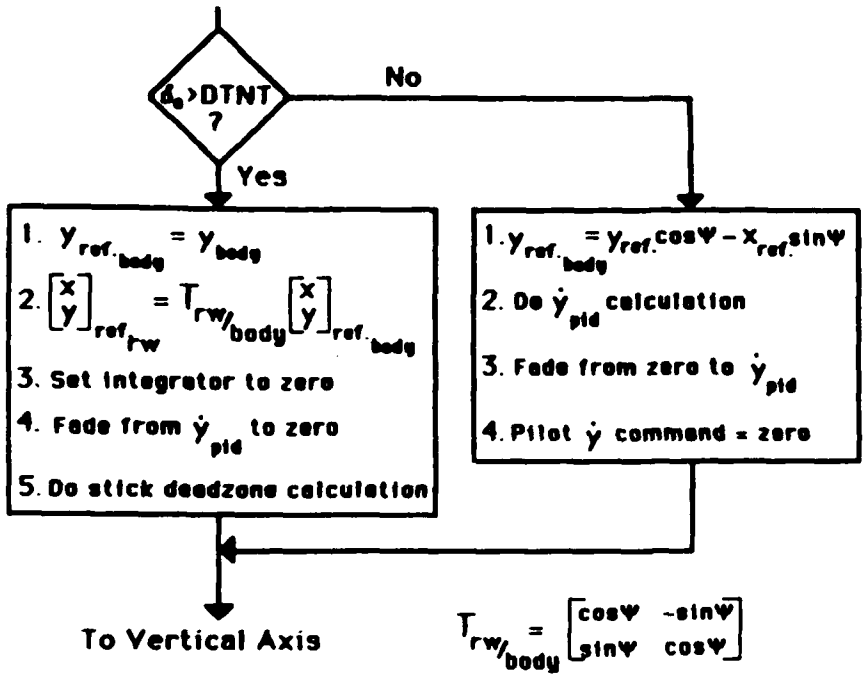


Figure 4.16: **Y Axis Transient-Free Switching Logic.** The switching logic for the x and y axes was identical. The best detent was .25 inches.

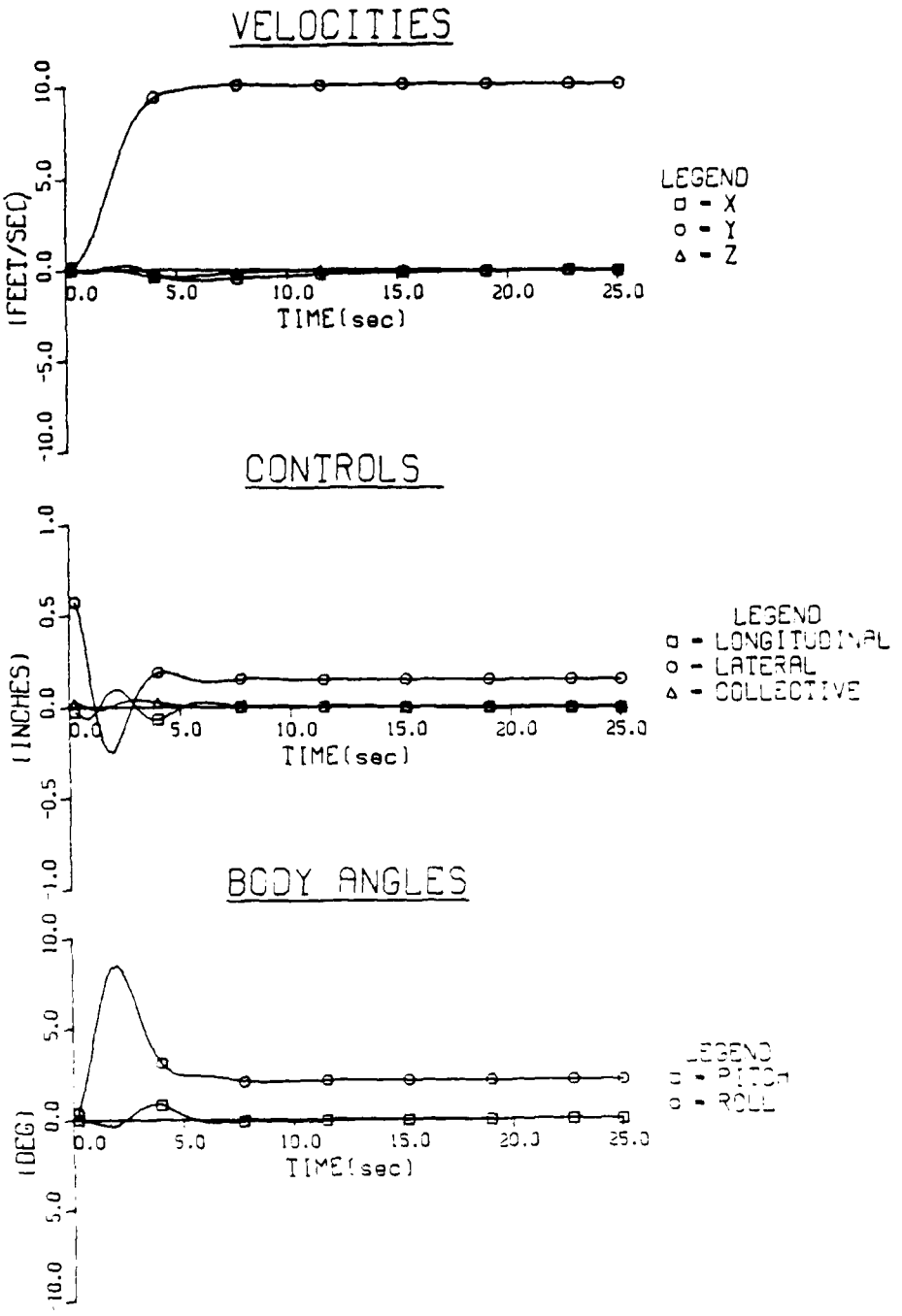


Figure 4.26: **Hover Side Velocity Step Command in Simulation.** The response is nearly identical to the original design of Figure 4.10.

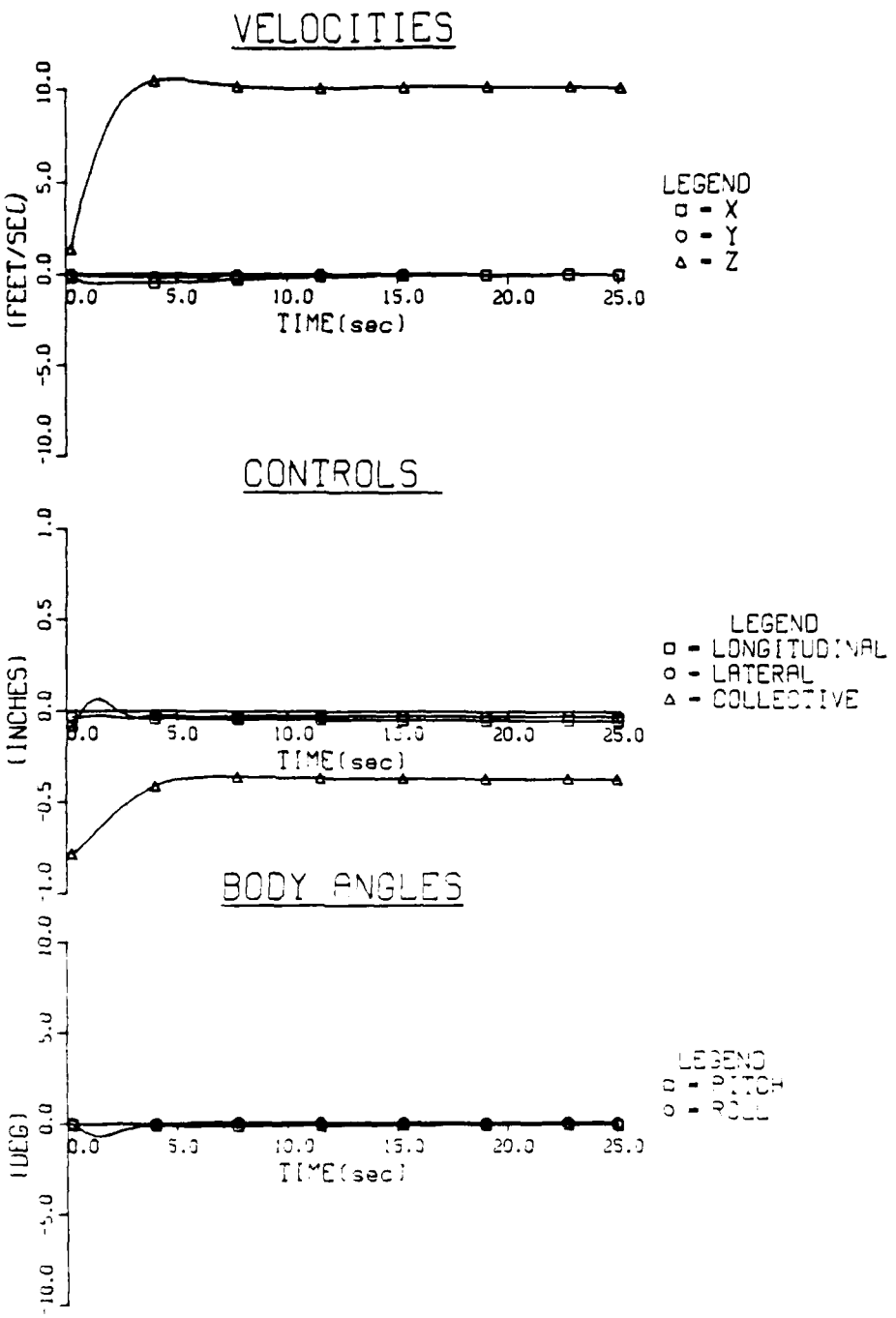


Figure 4.27: **Hover Vertical Velocity Step Command in Simulation.** The response is nearly identical to the original design of Figure 4.11.

4.8. Summary of Results of the Hover Controller Design

As with the longitudinal CAS, the discussion of results is separated into two groups:

- the effectiveness of the methodology
- the flight test results

The hover controller emphasized the usefulness of the design methodology for a more complicated control system. To have used classical incremental loop closures to do this design would probably have taken longer or would have required more specific experience in helicopter control systems than I had. This task also showed the advantage of using a modern control inner loop to modify the open loop plant in such a way as to increase the physical intuition for the design engineer. The increased physical intuition made classically designed outer loops simpler. In this case, the plant was changed from control motion in, measurement out to desired output in, actual output out. This change simplified the selection of the outer loop control structure and made the outer loop gains more intuitive. Figure 4.3 showed these advantages. This task also emphasized the relative speed and ease with which design iterations can be made on MIMO systems. When the first design of the hover controller was found unacceptable in flight, the redesign described in Section 4.7.3 was done in only 2 - 3 days which avoided delays in the flight testing. As with the longitudinal CAS design, the analysis tools developed to use the methodology (described in the various appendices) were sufficient but their

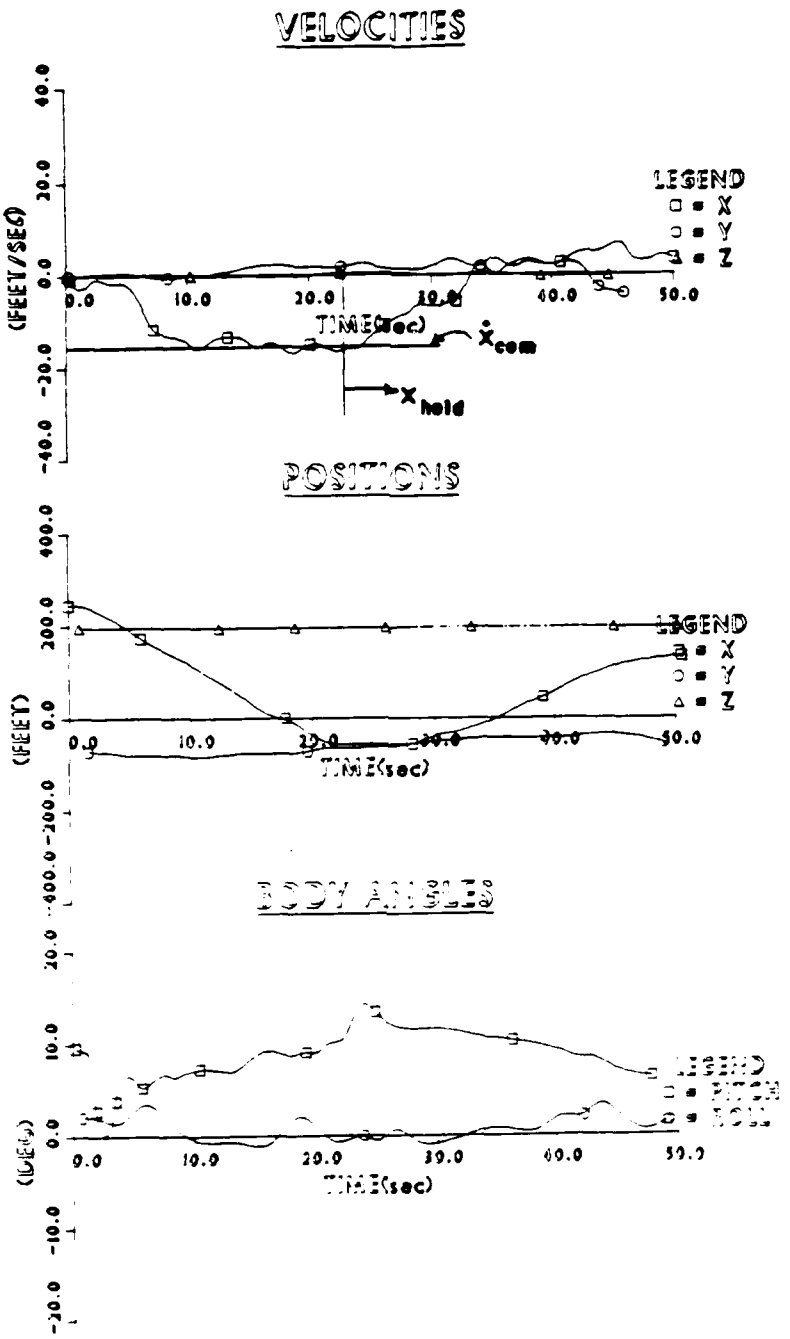


Figure 4.28: **Final $\dot{x}$ Flight Response.** The instability in pitch (Figure 4.18) and the pitch roll coupling (Figure 4.19) are gone but the pitch angle damping is still less than the simulation response of Figure 4.25.

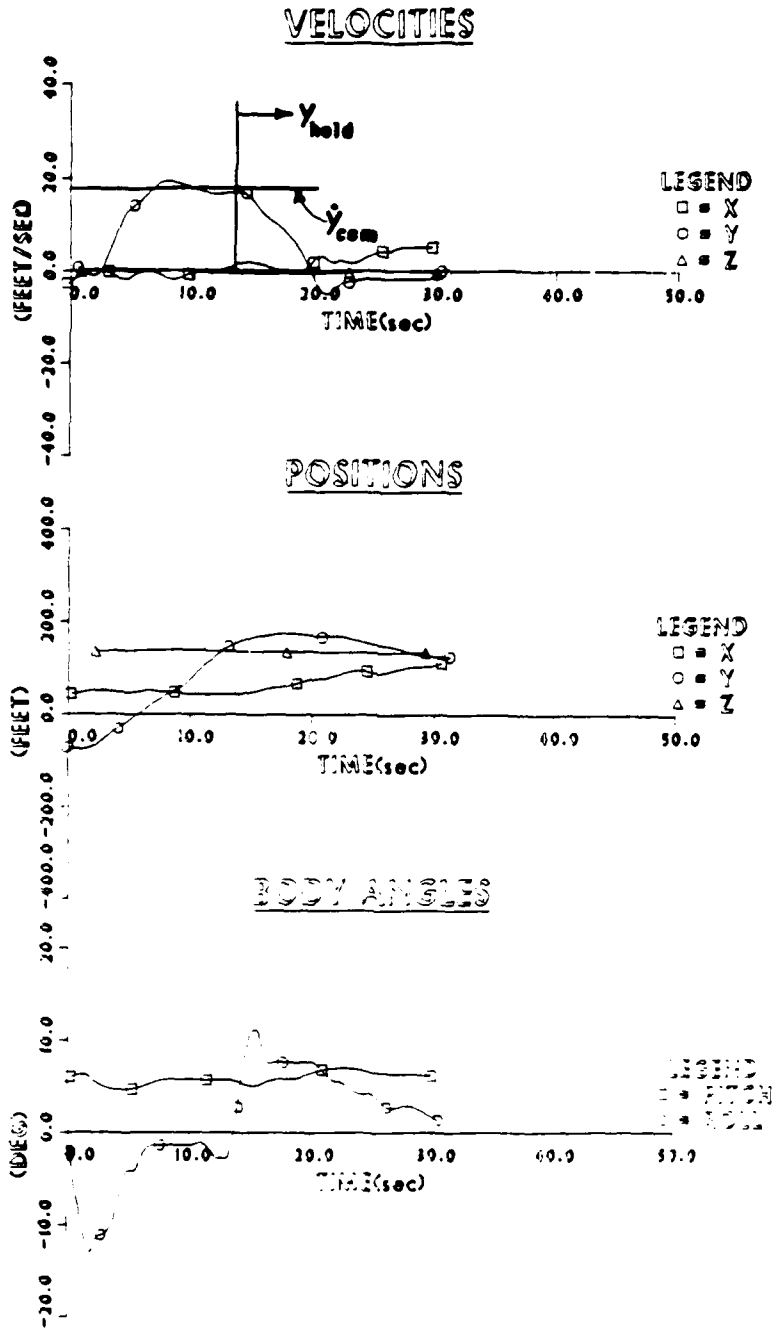


Figure 4.29: **Final $\dot{y}$ Flight Response.** The $\dot{y}$ response is little changed from the first flight test results of Figure 4.20.

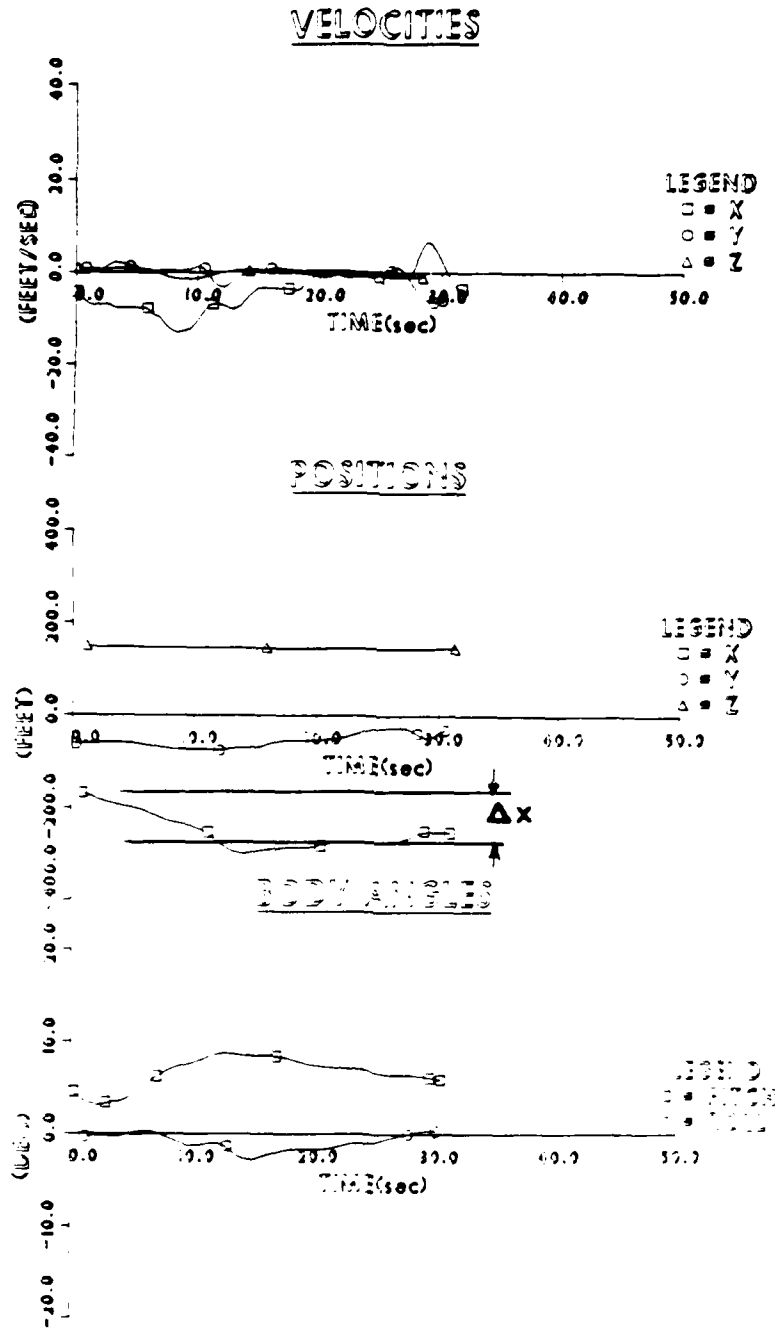


Figure 4.30: **Final Hover Forward Position Step Command in Flight.** The performance is much improved from Figure 4.22 but the damping of $\approx .4$ is still not good enough for operational use.

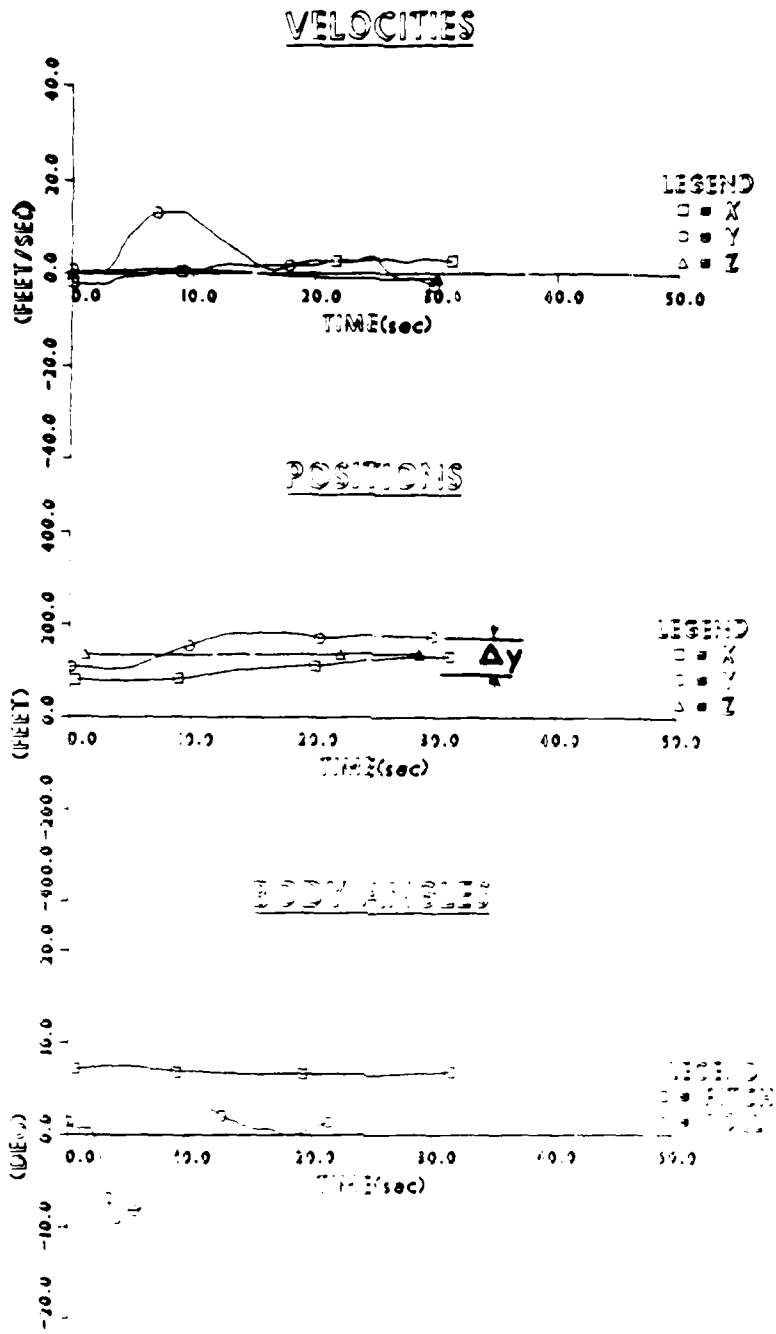


Figure 4.31: **Hover Lateral Position Step Command in Flight.** The lateral position response is adequate and little changed from the first tests of Figure 4.23.

"user friendliness" needed improvement.

In flight test, the hover controller was fairly successful. Many advanced helcopter or VTOL airplane designs call for a translational velocity control(TVC) system such as was tested here. Normally the evaluations of these concepts seldom leave the simulators to address the hardware and software difficulties of flight implementation. This test reemphasized two of the important difficulties of TVC systems, the inertial position/velocity sensor and the human factors involved. The primary contribution of this work was the development of a flexible TVC system where these type issues can be studied. Specifically, this system showed good velocity command performance in all three axes, excellent hold performance in the lateral and vertical axes, and marginally acceptable hold performance longitudinally. The switching logic worked well from a control viewpoint but the pilots who flew the system commented on the need for a better indication of switching from hold mode to velocity command mode in each axis.

Chapter 5.

Conclusions

The conclusions are separated into those applicable to the design methodology and those associated with the flight tests.

5.1. Methodology

- The process of scaling and using the modal input/output measures is an effective way to reduce the order of the compensator.
- The scaled block minimal realization of the compensator is useful in identifying unimportant measurements and controls.
- The decoupling feedforward matrix depends heavily on an accurate model so practical designs will usually require some sort of integral control. For very poor models, integral control alone should be used for implementing output commands.
- The software tools were adequate for application of the methodology.
- Both "modern" and "classical" control techniques are important for MIMO control system design. The specific application determines the appropriate techniques to use. In this research, the use of a modern control inner loop

with classically designed outer loops was a useful approach for the hover controller.

5.2. Flight Tests

- Although requiring some change of pilot technique (retraining), the decoupled velocity and climb rate controller was well received by the pilots who flew it.
- The hover controller performed adequately as a translational velocity command system, had good position hold capability in the vertical and lateral axes, but its hold performance in the longitudinal direction was marginal.
- Integral control was crucial to achieve decoupled control for both the cruise and hover control systems.

5.3. Lessons Learned

Finally, two "lessons learned" (or relearned) during this research should be emphasized, even though they may seem obvious. First, there is no substitute for experience. For this methodology, experience was important in:

- selecting the correct units for scaling the dynamic system
- determining which gains are "small" for compensator simplification
- selecting outputs and their weightings in the optimization using the ROPTSYS and RSANDY computer programs
- selecting scale factors for fixed-point scaling, Appendix E

- selecting stick and collective lever gains for the pilot
- determining the structure of the integral control loops

Also, experience in use of the methodology itself, especially the design tools, was critical. The hover controller, though much more complicated, took about as much time to design as the longitudinal CAS. The other lesson is that the design of the control logic is often the easiest and fastest step in building an operational control system. Most of the work is spent on:

- software design, coding, and testing
- hardware modifications and testing
- ground based closed loop testing

Chapter 6.

Recommendations for Further Research

The CH-47 research helicopter at Ames is a very flexible test vehicle and is being improved by the addition of a floating point digital computer programmable in a higher order language. With this improvement, a number of potential research projects should be considered:

- Parameter identification to improve the existing models used for design and simulation
- Refinement of the two designs presented here and pilot evaluation in a more realistic setting such as instrument landing
- Outer loop guidance work (Microwave Landing System, 4-dimensional navigation, etc.) using these inner loops
- Application of singular value LQG-LTR (Linear Quadratic Gaussian - Loop Transfer Recovery) to account for unmodeled rotor dynamics

One difficulty in applying the methodology was the poor convergence characteristics of the first order gradient algorithm in the RSANDY program. A second order technique to speed convergence would be an important improvement to the program. Another possibility for research is finding a way of commanding a system *without exciting* all the closed-loop modes similar to the method described in

Chapter 3 for the case of the full-order compensator.

Saberi has shown a technique for calculating helicopter stability derivatives during low speed flight near the ground.[18] This research vehicle is an excellent testbed for validating these derivatives.

References

1. McRuer,D., Ashkenas, I., and Graham, D., *Aircraft Dynamics and Automatic Control*, Princeton University Press, Princeton, New Jersey, 1973.

2. Bryson,A.E., Notes from the "Advanced Stability and Control of Flight Vehicles" course at Stanford(AA271b), 1983.

3. Ly,U.L., "A Design Algorithm for Robust Low-Order Controllers", Stanford University, SUDAAR 536, November 1982.

4. Bryson,A.E. and Ho,Y., *Applied Optimal Control*, Hemishpere Publishing Corporation, New York, 1975.

5. Kailath,T., *Linear Systems*, Prentice-Hall,Inc., Englewood Cliffs, New Jersey, 1980.

6. Franklin,G.F. and Powell,J.D., *Digital Control of Dynamic Systems*, Addison-Wesley Publishing Company, Menlo Park, California, 1981.

7. Bernard,D., "Control System Design for Lightly Coupled Space Structures", PhD Dissertation, Stanford University, September 1984.

8. Ly,U.L., "Optimal Robust Low-Order Controller Design, User Guide for the Program SANDY", Stanford University, 1982.

Appendix D.

Set Point Design

This appendix derives the feedforward matrix which commands a dynamic system, including compensator, to a new equilibrium. This matrix turns desired outputs into the steady-state controls needed to achieve the outputs. The restrictions are that these outputs (or new operating point) be physically realizable and that the number of controls be equal to the number of outputs.

Consider the following dynamic system:

$$\begin{aligned} \dot{x} &= Fx + Gu \\ y_s &= H_s x + D_{su} u \\ \dot{z} &= Az + By_s + G_z u_c \\ u &= Cz + Dy_s + u_c \end{aligned} \tag{D.1}$$

where:

x- plant states

y_s- measurements

z- compensator states

u- controls

The general form of the transformation is:

$$T = \begin{bmatrix} T_1 & 0 & \dots & 0 \\ 0 & T_2 & 0 & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & \frac{1}{c_l} \end{bmatrix}$$
$$T^{-1} = \begin{bmatrix} T_1^{-1} & 0 & \dots & 0 \\ 0 & T_2^{-1} & 0 & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & c_l \end{bmatrix} \qquad (C.11)$$

where:

$$T_i = \frac{\begin{bmatrix} c_{l,2i}(f_i g_i - e_i h_i) & e_i c_{l,2i} - b_i c_{l,2i-1} \\ -c_{l,2i-1}(f_i g_i - e_i h_i) & g_i c_{l,2i-1} - h_i c_{l,2i-1} \end{bmatrix}}{-f_i {c_{l,2i-1}}^2 + (e_i - h_i) c_{l,2i-1} c_{l,2i} + g_i {c_{l,2i}}^2} \qquad (C.12)$$

i– the i^{th} complex mode

l– row index corresponding to the largest value of $c_1^2 + c_2^2$ for that mode's double column, or the largest value of c_1^2 for the a real mode

A listing of a FORTRAN subroutine, MINCOM, which does this transformation is shown in Appendix L.

The "*0 1*" in the C_{min} matrix results from scaling the system by the largest values in the C double columns.

Consider a second order system in general form:

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} y$$
$$u = \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \tag{C.6}$$

The desired transformation will put this system into the following form:

$$\begin{bmatrix} \dot{z}_1' \\ \dot{z}_2' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ a_1 & a_2 \end{bmatrix} \begin{bmatrix} z_1' \\ z_2' \end{bmatrix} + \begin{bmatrix} b_1' \\ b_2' \end{bmatrix} y$$
$$u = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} z_1' \\ z_2' \end{bmatrix} \tag{C.7}$$

The similarity transformation requires the two systems to have identical eigenvalues, that is:

$$|sI - A| = |sI - A_{min}|$$
$$\Downarrow$$
$$a_1 = fg - eh \tag{C.8}$$
$$a_2 = e + h$$

Introducing the transformation matrix and expanding:

$$TA_{min} = AT$$
$$C_{min} = CT$$
$$\begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ fg - eh & e + h \end{bmatrix} = \begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \tag{C.9}$$
$$\begin{bmatrix} 0 & 1 \end{bmatrix} = \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix}$$

where $c_1^2 + c_2^2$ has the largest magnitude of any row pair in double column associated with the mode being made minimal. Solving the equations above we have the desired transformation:

$$T = \frac{\begin{bmatrix} c_2(fg - eh) & ec_2 - bc_1 \\ -c_1(fg - eh) & gc_2 - hc_1 \end{bmatrix}}{-fc_1^2 + (e - h)c_1c_2 + gc_2^2} \tag{C.10}$$

where:

$$A = \begin{bmatrix} \star & \star & 0 & 0 & \dots \\ \star & \star & & & \\ 0 & & \star & \star & 0 & 0 \\ & & \star & \star & & \\ 0 & & 0 & \ddots & 0 \\ \vdots & & 0 & 0 & \star \end{bmatrix}, r \times r \qquad (C.2)$$

$$B = [Full], r \times p$$

$$C = [Full], m \times r$$

We want a transformation to a new form:

$$\dot{z}' = A_{min}\, z' + B_{min}\, y$$
$$u = C_{min}\, z' \qquad (C.3)$$

where:

A_{min}- minimal form of A

B_{min}- minimal form of B

C_{min}- minimal form of C

$$A_{min} = \begin{bmatrix} 0 & 1 & & 0 & \dots \\ a_1 & a_2 & & & \\ & 0 & & 0 & 1 & \dots \\ & & & a_1 & a_2 & \\ & \vdots & & \vdots & & \ddots \end{bmatrix}, n \times n$$

$$B_{min} = \begin{bmatrix} \star & \star & \dots \\ \star & \star & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}, n \times p \qquad (C.4)$$

$$C_{min} = \begin{bmatrix} \star & \star & 0 & 1 & \dots \\ 0 & 1 & \star & \star & \dots \\ \star & \star & \star & \star & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, m \times n$$

$$z = Tz'$$
$$A_{min} = T^{-1}AT$$
$$B_{min} = T^{-1}B \qquad (C.5)$$
$$C_{min} = CT$$

Appendix C.

Minimal Realizations

The design methodology described in Chapter 2 used minimal realizations in two places. The first was when the ROPTSYS computer program displayed the compensator in minimal form to be better suited for the optimization in the RSANDY program. This eliminated redundant parameters which could cause trouble in the RSANDY gradient search procedures. The second use of a minimal realization came when the discrete compensator was transformed to minimal form for computational efficiency. In the first case, the transformation was from arbitrary form to block modal form then to block minimal form. The second was from an arbitrary 2×2 block form to the block minimal form. The derivation is shown for an arbitrary 2×2 system then expanded for any order.

Given the following form of the dynamic system:

$$\begin{aligned} \dot{z} &= Az + By \\ u &= Cz \end{aligned} \tag{C.1}$$

By the duality property of regulators and estimators (Figure B.1), these gains, $K^T = (R + NQN^T)^{-1}(NQ\Gamma^T + H_m P)$, are determined using the randomly disturbed equations of motion:

$$\begin{aligned} \dot{x} &= Fx + \Gamma w \\ y_m &= H_m + Nw + v \end{aligned} \tag{B.12}$$

where

Q– noise spectral density matrix of plant distrubances, w

R– noise spectral density matrix of measurement noise, v

Regulator	F	G	H	L	A	B	S	C
Estimator	F^T	$H_m{}^T$	Γ^T	N^T	Q	R	P	K^T

Figure B.1: **Duality Between Regulators and Estimators.** This shows the property of duality which allows the use of the regulator results for design of an optimal estimator (a steady-state Kalman filter).

Adjoining the constraints (the equations of motion) to form the Hamiltonian:

$$H = \mathcal{L} + \lambda^T(Fx + Gu) \tag{B.5}$$

Recalling the optimality conditions:

$$\dot{\lambda}^T = \frac{\partial H}{\partial x} \tag{B.6}$$

$$0 = \frac{\partial H}{\partial u} \tag{B.7}$$

Introducing the system equations, $\dot{x} = Fx + Gu$, and expanding the optimality equations, we have the Euler-Lagrange equations (here in matrix form):

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} F - G(B + L^T AL)^{-1} L^T AM & -G(B + L^T AL)^{-1} G^T \\ -M^T AM - M^T AL(B + L^T AL)^{-1} L^T AM & -F - M^T AL(B + L^T AL)^{-1} G^T \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \tag{B.8}$$

As shown in *Bryson and Ho* [4] or *Franklin and Powell* [6], the solution to these equations is $\lambda = Sx$ where $S = \Lambda_- \mathcal{X}_-^{-1}$. Λ_- and $\mathcal{X}_-$ are the submatrices of the eigenvector matrix of the Hamiltonian matrix associated with eigenvalues having negative real values, i.e.

$$\begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathcal{X}_- & \mathcal{X}_+ \\ \Lambda_- & \Lambda_+ \end{bmatrix} \begin{bmatrix} \xi_- \\ \xi_+ \end{bmatrix} \tag{B.9}$$

With this solution for λ, the optimal steady state control, u, can be expressed as a linear combination of the state variables, x:

$$\begin{aligned} u &= Cx \\ C &= (B + L^T AL)^{-1}(L^T AM + G^T S) \end{aligned} \tag{B.10}$$

The same approach applies to finding the estimator gains, K, of the equation:

$$\dot{\hat{x}} = F\hat{x} + Gu + K(y_m - H_m\hat{x}) \tag{B.11}$$

Appendix B.

Optimal Compensator Design

The design methodology described in Chapter 2 uses an optimal full order compensator as the starting point. This appendix summarizes the derivation of the optimal compensator. From Hall and Bryson, we see that to design a set of regulator gains which minimize a quadratic performance index, we minimize the Hamiltonian with respect to the control.[17] In this case, the performance index includes the control in the output, thus enabling the weighting of state rates(accelerations). This is essential in aerospace applications where vehicle acceleration is an important parameter in the design and analysis of the control system. Starting with the modified performance index:

$$J = \int_0^\infty \mathcal{L} dt = \int_0^\infty \frac{1}{2}(y^T A y + u^T B u) dt \qquad (B.1)$$

where:

$$y = Mx + Lu \qquad (B.2)$$

$$y^T = x^T M^T + u^T L^T \qquad (B.3)$$

Expanding the integrand of J:

$$\mathcal{L} = \frac{1}{2}\left[x^T M^T A M x + x^T M^T A L u + u^T L^T A M x + u^T (L^T A L + B) u\right] \qquad (B.4)$$

σ- standard deviation of the noise variable
T_c- correlation time of the noise

With these transformations, the resulting compensator will use scaled measurements to calculate a scaled control signal. If we want to use the compensator in the physical system, we need only unscale the gain matrices. The compensator based on the scaled variables is:

$$\begin{aligned} \dot{z} &= \bar{A}z + \bar{B}\bar{y}_s \\ \bar{u} &= \bar{C}z + \bar{D}\bar{y}_s \end{aligned} \tag{A.8}$$

We unscale the system by replacing scaled vectors $\bar{u}$ and $\bar{y}_s$ with their unscaled equivalents $\bar{u} = {T_c}^{-1}u$ and $\bar{y}_s = {T_m}^{-1}y_s$:

$$\begin{aligned} \dot{z} &= \bar{A}z + \bar{B}\,{T_m}^{-1}\,y_s \\ \bar{u} &= T_c\bar{C}z + \bar{D}\,{T_m}^{-1}\,y_s \end{aligned} \tag{A.9}$$

Now the compensator uses actual (unscaled) measurements and gives unscaled control signals as outputs.

To make the scaling process consistent, I've listed some rules of thumb below:

- Scale the matrices consistently; for example, if a measurement is also a state, use the same units.
- Scale intermediate state variables in an actuator model the same as the control itself. For example, if we have a first order actuator model, $\dot{u}_a = -au_a + au_c$, then scale the actuator position state(u_a), its rate($\dot{u}_a$), and the command(u_c) identically.
- Similarly, if sensor noise filters are included in the plant model, then these noise filter states should be scaled the same as the measurements they filter.

turbances:

$$\begin{aligned} y_s &= T_m \bar{y}_s \\ y_c &= T_p \bar{y}_c \\ w &= T_d \bar{w} \\ v &= T_m \bar{v} \end{aligned} \tag{A.4}$$

With these transformations, the scaled dynamic system is:

$$\begin{aligned} \dot{\bar{x}} &= \bar{F}\bar{x} + \bar{G}\bar{u} + \bar{\Gamma}\bar{w} \\ \bar{y}_s &= \bar{H}_s\bar{x} + \bar{D}_{su}\bar{u} + \bar{N}\bar{w} + \bar{v} \\ \bar{y}_c &= \bar{H}_c\bar{x} + \bar{D}_{cu}\bar{u} \end{aligned} \tag{A.5}$$

where:

$$\begin{aligned} \bar{F} &= T_s^{-1} F T_s \\ \bar{G} &= T_s^{-1} G T_c \\ \bar{\Gamma} &= T_s^{-1} \Gamma T_d \\ \bar{H}_s &= T_m^{-1} H_s T_s \\ \bar{D}_{su} &= T_m^{-1} D_{su} T_c \\ \bar{N} &= T_m^{-1} N T_d \\ \bar{H}_c &= T_p^{-1} H_c T_s \\ \bar{D}_{cu} &= T_p^{-1} D_{cu} T_c \\ \bar{Q} &= T_d^{-1} Q T_d^{-1} \\ \bar{R} &= T_m^{-1} R T_m^{-1} \end{aligned} \tag{A.6}$$

The scaled power spectral density matrices were derived using the approximation:

$$PSD \approx 2\sigma^2 T_c \tag{A.7}$$

where:

F- plant dynamics matrix, $n \times n$

G- control distribution matrix, $n \times m$

Γ- plant disturbance distribution matrix, $n \times m'$

H_s- state to measurement distribution matrix, $p \times n$

D_{su}- control to measurement distribution matrix, $p \times m$

N- plant disturbance to measurement distribution matrix, $p \times m'$

H_c- state to output distribution matrix, $p' \times n$

D_{cu}- control to output distribution matrix, $p' \times m$

Q- plant disturbance spectral density matrix, $m' \times m'$

R- sensor noise spectral density matrix, $p \times p$

The scaling process continues by describing the changes of units on the states, controls, etc. as simple transformations. For instance, if we want new states, $\bar{x}$, and new controls, $\bar{u}$, to be $s_{x_1}x_1$, $s_{x_2}x_2$, ... $s_{x_n}x_n$ and $s_{u_1}u_1$, $s_{u_2}u_2$, ... $s_{u_n}u_n$, then we can define scaling (also similarity) transformations:

$$\begin{aligned} x &= T_s\bar{x} \\ u &= T_c\bar{u} \end{aligned} \tag{A.2}$$

where:

$$\begin{aligned} T_s &= \begin{bmatrix} \frac{1}{s_{x_1}} & & & \\ & \frac{1}{s_{x_2}} & & \\ & & \ddots & \\ & & & \frac{1}{s_{x_n}} \end{bmatrix} \\ T_c &= \begin{bmatrix} \frac{1}{s_{u_1}} & & & \\ & \frac{1}{s_{u_2}} & & \\ & & \ddots & \\ & & & \frac{1}{s_{u_n}} \end{bmatrix} \end{aligned} \tag{A.3}$$

Using an identical procedure, we scale the outputs, measurements, and plant dis-

Appendix A.

Engineering Scaling

This appendix derives engineering scaling equations used in the ROPTSYS computer program. This process transforms the model of the physical system into a "similar" model where the units of the variables have changed. Similar means the eigenvalues of the system are not changed by the transformation to the new coordinates. As described in section 2.2, the new units are chosen to make the new variables of the dynamic system of equal importance to the design engineer. The process begins with the linear model shown below:

$$\begin{aligned} \dot{x} &= Fx + Gu + \Gamma w \\ y_s &= H_s x + D_{su} u + N w + v \\ y_c &= H_c x + D_{cu} u \\ J &= \int_0^\infty (y_c^T A y_c + u^T B u)\, dt \end{aligned} \tag{A.1}$$

where:

- x- system states, $n \times 1$
- z- compensator states, $r \times 1$
- u- controls, $m \times 1$
- w- plant disturbances, $m' \times 1$
- y_s- sensor measurements, $p \times 1$
- y_c- weighted outputs, $p' \times 1$
- v- sensor noise, $p \times 1$

ber, 1971.

18. Saberi,H.A.,"Ground Effect on Helicopter Aerodynamics and Stability",PhD Dissertation, Stanford University, October 1984.

9. Bryson,A.E., Notes from the "Optimal Estimation and Control Logic in the Presence of Noise" course at Stanford(AA278b), 1984.

10. Kelly,J.R., Niessen,F.R.,Garren,J.F., and Abbott,T.S., "Description of the VTOL Approach and Landing Technology(VALT) CH-47 Research System", NASA TP 1436, August 1979.

11. Ostroff,A.J., Downing,D.R., and Rood,W.J., "A Technique Using a Nonlinear Helicopter Model for Determining Trims and Derivatives", NASA TN D-8159, May 1976.

12. Chen,R.T.N. and Hindson,W.S., "Influence of Rotor and Other High-Order Dynamics on Helicopter Flight Control System Bandwidth", a paper to be presented at the International Conference on Rotorcraft Basic Research, Research Triangle Park, NC, February 1985.

13. Hutto,A.J., "Flight Test Report on the Heavy-Lift Helicopter Flight Control System", 31st Annual National Forum of the American Helicopter Society, May 1975.

14. Etkin,B., *Dynamics of Atmospheric Flight*, John Wiley and Sons,Inc., Toronto, 1972.

15. Valkenburg,M.E., *Introduction to Modern Network Synthesis*, John Wiley and Sons,Inc., New York, 1964.

16. Gardner,B.E.,"Feedforward/Feedback Control Logic for Robust Target Tracking", PhD Dissertation, Stanford University, December 1984.

17. Hall,W.E. and Bryson,A.E., "Optimal Control and Filter Synthesis by Eigenvector Decomposition", SUDAAR No. 436, Stanford University, Novem-

If $DD_{eu} = 0$, then these equations can be rewritten as:

$$\begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} F + GDH_e & GC \\ B(I + D_{eu}D)H_e & A + BD_{eu}C \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} G \\ BD_{eu} + G_z \end{bmatrix} u_c \qquad (D.2)$$

Defining $x_T = \begin{bmatrix} x \\ z \end{bmatrix}$, we can rewrite the equations above as:

$$\dot{x_T} = F_T x_T + G_T u_c \qquad (D.3)$$

Expressing the desired outputs, y_D, as a linear combination of x_T and u_c we have:

$$y_D = H_D x_T + L_D u_c \qquad (D.4)$$

At steady state, $\dot{x_T} = 0$, and the two previous equations become:

$$\begin{bmatrix} F_T & G_T \\ H_D & L_D \end{bmatrix} \begin{bmatrix} x_T \\ u_c \end{bmatrix}_{ss} = \begin{bmatrix} 0 \\ y_D \end{bmatrix} \qquad (D.5)$$

inverting:

$$\begin{aligned} \begin{bmatrix} x_T \\ u_c \end{bmatrix}_{ss} &= \begin{bmatrix} F_T & G_T \\ H_D & L_D \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ y_D \end{bmatrix} \\ &= \begin{bmatrix} \star & M \\ \star & N \end{bmatrix} \begin{bmatrix} 0 \\ y_D \end{bmatrix} \end{aligned} \qquad (D.6)$$

The steady-state controls are $u_c = Ny_D$ and new equilibrium state vector is $x_T = My_D$ where:

$$\begin{bmatrix} \star & M \\ \star & N \end{bmatrix} = \begin{bmatrix} F + GDH_e & GC & G \\ B(I + D_{eu}D)H_e & A + BD_{eu}C & BD_{eu} + G_z \\ H_D & & L_D \end{bmatrix}^{-1} \qquad (D.7)$$

This last equation is used by the SETPNT program, Appendix L, to calculate the N matrix.

Appendix E.

Fixed Point Scaling

This appendix describes the technique of scaling the analytical designs to run on the Sperry 1819A flight computer. The process is similar in principle to scaling engineering problems for an analog computer. This computer is an 18 bit fixed-point digital computer. There are two problems which must be considered when doing this scaling. The first is avoiding overflows (exceeding $2^{17} - 1$ during calculations) and the second is maintaining precision in the results. The procedure which follows handles both these potential problems.

Consider the compensator dynamic system:

$$\begin{aligned} \dot{z} &= Az + By \\ u &= Cz \end{aligned} \tag{E.1}$$

where y and u are in engineering units (not yet computer scaled). In the computer, these variable have computer scaling factors, K, such that $\dot{z}K_{\dot{z}}$, zK_z, yK_y, and uK_u have units of bits. For example, if $K_\theta = 500\frac{bits}{deg}$, then then 5 deg of θ is 2500 bits in the computer. In these computer scaled variables, the compensator appears:

$$\begin{aligned} [\dot{z}]\, K_{\dot{z}} &= K_A\, [A]\, \frac{K_{\dot{z}}}{K_z K_A}\, [z]\, K_z + K_B\, [B]\, \frac{K_{\dot{z}}}{K_y K_B}\, [y]\, K_y \\ [u]\, K_u &= K_C\, [C]\, \frac{K_u}{K_z K_C}\, [z]\, K_z \end{aligned} \tag{E.2}$$

where A, B, and C are in engineering units. The scale factors, K, for y and u are part of the computer environment (set up by the programmers of the original flight program) but we need to calculate $K_{\dot{z}}$ and K_z. This is simplified by the digital implemention; we have z_{k+1} and z_k rather than $\dot{z}$ and z. This means only one scale facter, K_z, is needed. To find this factor, first estimate the largest value that any z_{k+1} or z_k can achieve by finding the maximum single product in the matrix multiply, $[B]\,[y]_{max}$. For controls, measurements, and desired outputs, the maximum values can be set using engineering judgement and intuition. Since we also want precision in the z term, we select K_z so that z_{max} uses all of the 18 bits available:

$$K_z = \frac{2^{17}}{z_{max}} \tag{E.3}$$

where z_{max} is rounded up to the next power of 2. 2^{17} is used since the largest negative number expressed in 18 bits is -2^{17}.

With the K_z term, the problem of overflow is solved. Now we need only ensure that precision is maintained in the calculations by choosing the additional scale factors, K_A, K_B, and K_C that scale the elements of the A, B, and C matrices. Making use of all 18 bits, we can find these scale factors in the same way as the K_z term above was calculated:

$$\begin{aligned} K_A &= \frac{2^{17}}{a_{max}\frac{K_z}{K_z}} \\ K_B &= \frac{2^{17}}{b_{max}\frac{K_z}{K_y}} \\ K_C &= \frac{2^{17}}{c_{max}\frac{K_z}{K_u}} \end{aligned} \tag{E.4}$$

where a_{max}, b_{max}, and c_{max} are the maximum elements of the A, B, and C matrices which have been rounded up to the nearest power of 2.

One product from the matrix multiplies is:

$$(b_{ij}\frac{K_z}{K_y})K_B(y_jK_y) \qquad (E.5)$$

This number is included in the double precesion(36 bits) A register in the 1819A and is always less than $2^{35}-1$. We want to accumulate these double precision elements to get one element of By. Finally, we divide by K_A (or equivalently shift the A register) to regain the single precision inner product. This is done for each of the matrix multiplies.

The same approach is used to scale the feedforward matrix and the integral gain matrix used in the longitudinal CAS. The two BASIC computer programs, which do this scaling for the longitudinal CAS and for the hover controller, are listed in Appendix N along with example data files.

Appendix F.

CH-47 Research Helicopter

The helicopter used for this research is a highly modified version of the Boeing-Vertol CH-47 "Chinook" used by the U.S Army for cargo and troop transport. Figure F.1 shows the tandem rotor helicopter, which is operated by the NASA Ames Research Center. Reference 10 is a more complete description of this particular helicopter including the many modifications made to the basic CH-47. Below are listed some of the modifications and improved capabilities:

- Full authority, variable stability, fly-by-wire flight control system in all four axes.
- Programmable analog and digital computers capable of executing the control laws.
- Programmable force-feel system on the experimental pilot's stick.
- Flight instrumentation system capable of recording over 100 variables at 100 times per second.
- Operator's console for control of the experimental systems.
- Additional sensors: INS, radar altimeter, body-mounted accelerometers, improved air data sensors, numerous control position sensors, boom-mounted angle of attack and sideslip vanes, rate gyros
- Digital ground to air uplink capability

Figures F.2 and F.3, from reference 10, show the cabin layout in this experimental vehicle and a block diagram of the experimental control system.

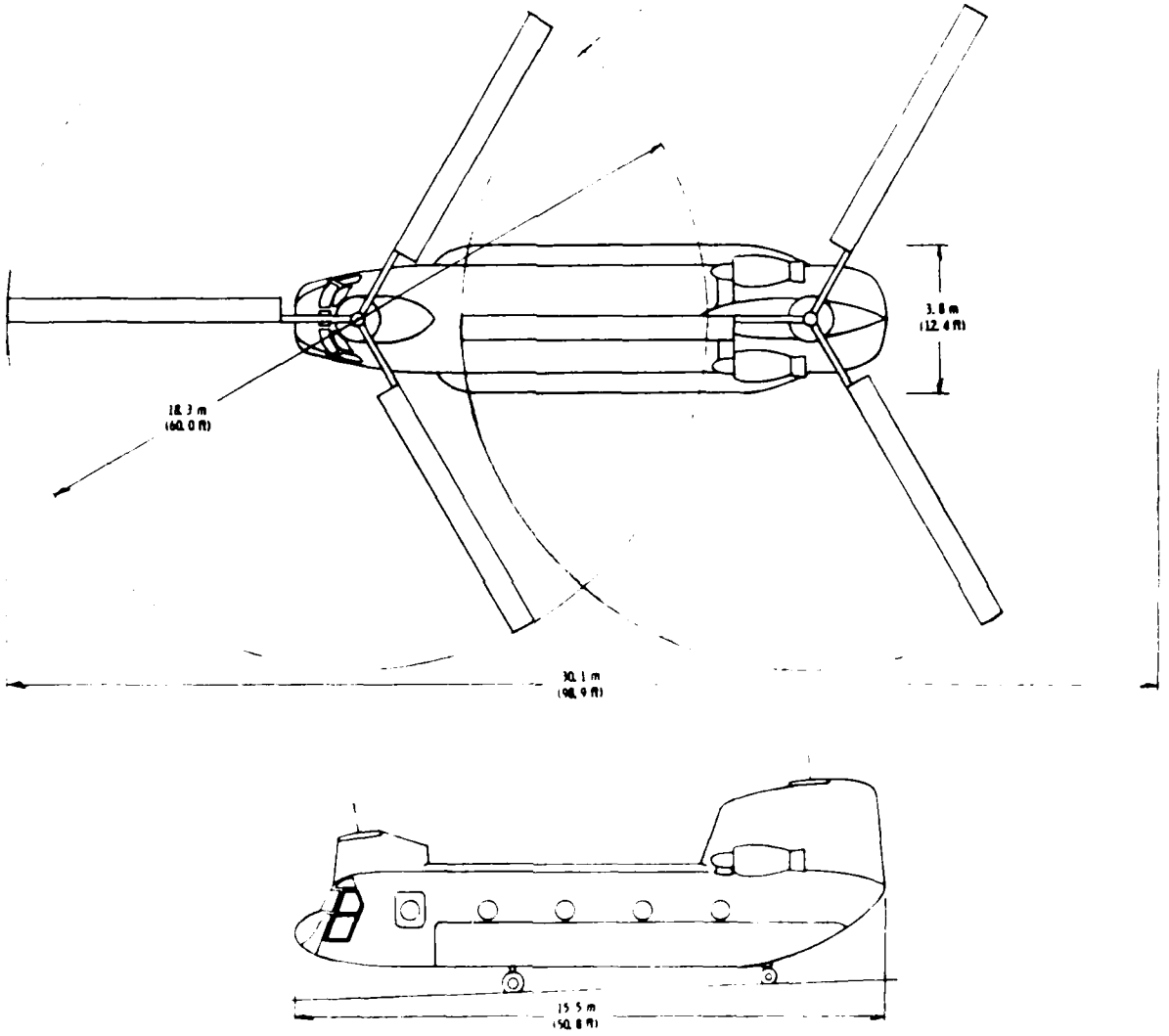


Figure F.1: **Boeing-Vertol CH-47 Chinook Helicopter.** The large tandem rotor helicopter is used operationally by the U.S. Army for cargo and troop transport. Maximum gross weight is 38000 pounds with typical operating wieght of 30000 pounds.

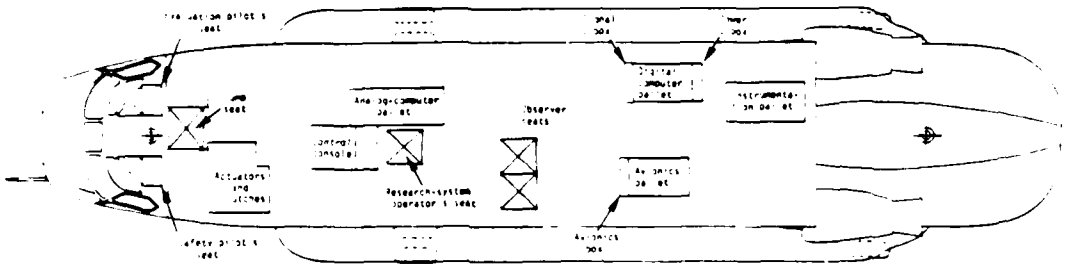


Figure F.2: **Cabin Layout.** The research helicopter requires a crew of 4; safety pilot, experimental pilot, research system operator, and crew chief.

Figure F.3: **Experimental Flight Systems.** The flexibility of the experimental system is emphasized here where we note the many interfaces between the various components.

Appendix G.

CH-47 Linear Models

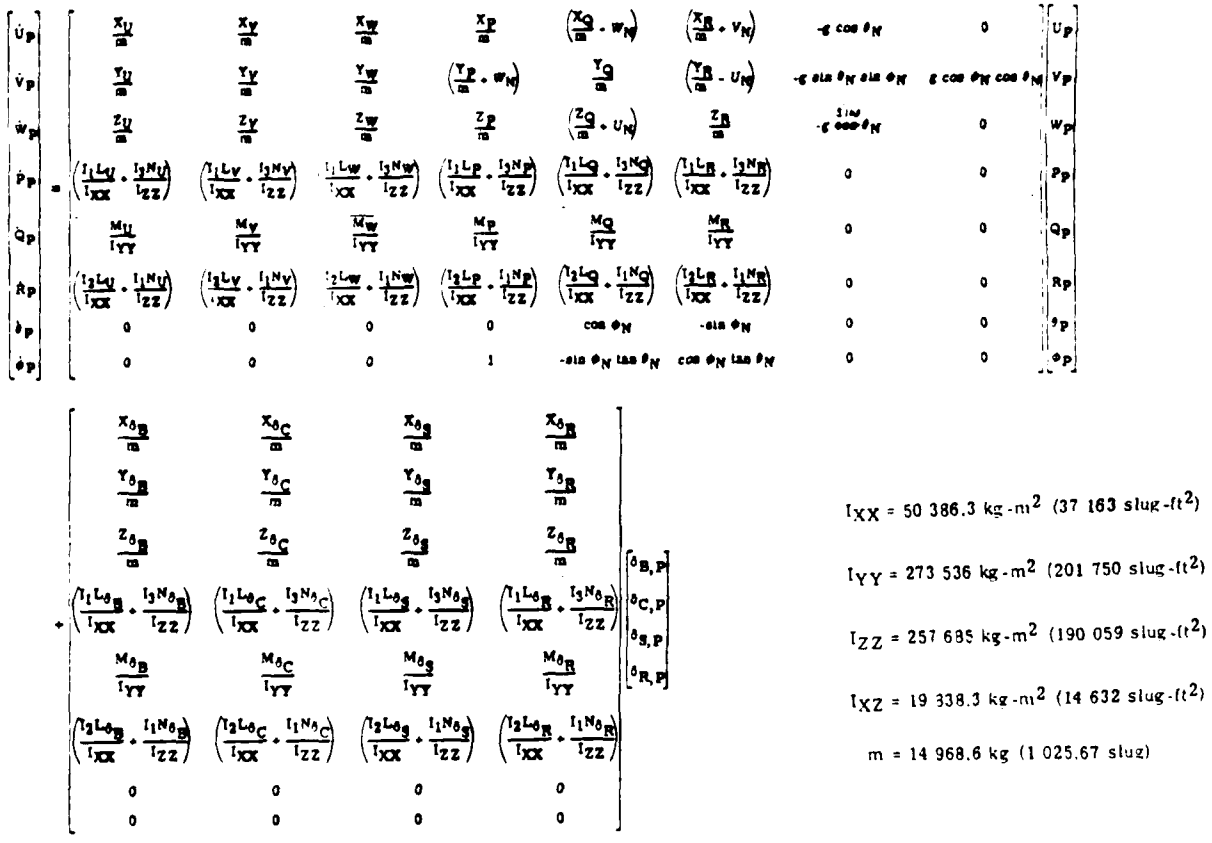
The models described in this appendix were calculated using the information from reference 11. The general forms for the decoupled 4^{th} order models and for the coupled 8^{th} order are shown in Figures G.1 and G.2. The linear models for the flight conditions related to this research are shown in Figures G.3 to G.16.

$$\begin{bmatrix} \dot{U}_P \\ \dot{W}_P \\ \dot{Q}_P \\ \dot{\theta}_P \end{bmatrix} = \begin{bmatrix} \frac{X_U}{m} & \frac{X_W}{m} & \left(\frac{X_Q}{m} - W_N\right) & -g\cos\theta_N \\ \frac{Z_U}{m} & \frac{Z_W}{m} & \left(U_N + \frac{Z_Q}{m}\right) & -g\sin\theta_N \\ \frac{M_U}{I_{YY}} & \frac{M_W}{I_{YY}} & \frac{M_Q}{I_{YY}} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} U_P \\ W_P \\ Q_P \\ \theta_P \end{bmatrix} + \begin{bmatrix} \frac{X_{\delta_B}}{m} & \frac{X_{\delta_C}}{m} \\ \frac{Z_{\delta_B}}{m} & \frac{Z_{\delta_C}}{m} \\ \frac{M_{\delta_B}}{I_{YY}} & \frac{M_{\delta_C}}{I_{YY}} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{B,P} \\ \delta_{C,P} \end{bmatrix}$$

$$\begin{bmatrix} \dot{P}_P \\ \dot{\phi}_P \\ \dot{R}_P \\ \dot{V}_P \end{bmatrix} = \begin{bmatrix} \left(\frac{I_1 L_P}{I_{XX}} + \frac{I_3 N_P}{I_{ZZ}}\right) & 0 & \left(\frac{I_1 L_R}{I_{XX}} + \frac{I_3 N_R}{I_{ZZ}}\right) & \left(\frac{I_1 L_V}{I_{XX}} + \frac{I_3 N_V}{I_{ZZ}}\right) \\ 1 & 0 & \tan\theta_N & 0 \\ \left(\frac{I_2 L_P}{I_{XX}} + \frac{I_1 N_P}{I_{ZZ}}\right) & 0 & \left(\frac{I_2 L_R}{I_{XX}} + \frac{I_1 N_R}{I_{ZZ}}\right) & \left(\frac{I_2 L_V}{I_{XX}} + \frac{I_1 N_V}{I_{ZZ}}\right) \\ \left(W_N + \frac{Y_P}{m}\right) & g\cos\theta_N & \left(\frac{Y_R}{m} - U_N\right) & \frac{Y_V}{m} \end{bmatrix} \begin{bmatrix} P_P \\ \phi_P \\ R_P \\ V_P \end{bmatrix}$$

$$+ \begin{bmatrix} \left(\frac{I_1 L_{\delta_S}}{I_{XX}} + \frac{I_3 N_{\delta_S}}{I_{ZZ}}\right) & \left(\frac{I_1 L_{\delta_R}}{I_{XX}} + \frac{I_3 N_{\delta_R}}{I_{ZZ}}\right) \\ 0 & 0 \\ \left(\frac{I_2 L_{\delta_S}}{I_{XX}} + \frac{I_1 N_{\delta_S}}{I_{ZZ}}\right) & \left(\frac{I_2 L_{\delta_R}}{I_{XX}} + \frac{I_1 N_{\delta_R}}{I_{ZZ}}\right) \\ \underbrace{Y_{\delta_S}}_{m} & \underbrace{Y_{\delta_R}}_{m} \end{bmatrix} \begin{bmatrix} \delta_{S,P} \\ \delta_{R,P} \end{bmatrix}$$

Figure G.1: **Longitudinal and Lateral 4^{th} Order Models.** The longitudinal model above was used in the design of the longitudinal CAS (Chapter 3).



$$I_{XX} = 50\,386.3 \text{ kg-m}^2 \ (37\,163 \text{ slug-ft}^2)$$

$$I_{YY} = 273\,536 \text{ kg-m}^2 \ (201\,750 \text{ slug-ft}^2)$$

$$I_{ZZ} = 257\,685 \text{ kg-m}^2 \ (190\,059 \text{ slug-ft}^2)$$

$$I_{XZ} = 19\,338.3 \text{ kg-m}^2 \ (14\,632 \text{ slug-ft}^2)$$

$$m = 14\,968.6 \text{ kg } (1\,025.67 \text{ slug})$$

where

$$I_1 = \frac{I_{XX} I_{ZZ}}{I_{XX} I_{ZZ} - I_{XZ}^2} \qquad I_2 = \frac{I_{XX} I_{XZ}}{I_{XX} I_{ZZ} - I_{XZ}^2} \qquad I_3 = \frac{I_{ZZ} I_{XZ}}{I_{XX} I_{ZZ} - I_{XZ}^2}$$

Figure G.2: **Coupled 8^{th} Order Model.** This model was used in the design of the hover controller, described in Chapter 4.

THIS IS DATA FOR ZDOT= 0 FT/MIN AND XDOT OR AIRSPEED=
60 KNOTS

	X	Y	Z	L	M	N
U	-0.02046	-0.00017	-0.06631	-0.00024	-0.00420	0.00030
V	0.00014	-0.07404	0.00487	-0.00548	-0.00006	-0.00009
W	0.03764	0.00330	-0.55118	0.00166	0.01764	-0.00026
DB	0.12688	0.51850	0.46734	0.00376	0.39113	0.02811
DC	0.42640	0.04854	-9.35989	-0.00864	0.15286	0.00552
DS	0.00003	1.11989	-0.00044	0.40554	0.00000	0.00887
DR	0.00004	-0.05298	0.00008	-0.13452	0.00011	0.19699
P	0.00380	-2.03591	0.21481	-0.81835	0.02270	-0.01663
Q	2.35790	0.00340	-1.17918	0.00117	-1.68183	-0.07923
R	-0.04676	-0.22199	0.28535	-0.06732	0.00267	-0.03912

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	-0.02046	0.03764	2.35790	-32.14647
2	-0.06631	-0.55118	99.02081	-1.85603
3	-0.00420	0.01764	-1.68183	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

	1	2
	0.12688	0.42640
	0.46734	-9.35989
	0.39113	0.15286
	0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.85072	0.00000	-0.08531	-0.00569
2	1.00000	0.00000	0.05774	0.00000
3	-0.08221	0.00000	-0.04570	-0.00053
4	-2.03591	32.14647	%-100.42200	-0.07404

LATERAL G-MATRIX IS:

	1	2
	0.42183	-0.05875
	0.00000	0.00000
	0.04139	0.19246
	1.11989	-0.05298

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	-0.02046	-0.00014	0.03764	0.00380	2.35790	-0.04676	-32.14647	0.00000
2	-0.00017	-0.07404	0.00330	-2.03591	0.00340	%-100.42200	0.00902	32.14
3								
4	-0.06631	0.00487	-0.55118	0.21481	99.02081	0.28535	-1.85603	0.00
5	-0.00042	-0.00589	0.00157	-0.85072	-0.07096	-0.08531	0.00000	0.00
6	-0.00420	-0.00006	0.01764	0.02270	-1.68183	0.00267	0.00000	0.00
7	-0.00009	-0.00053	-0.00024	-0.08221	-0.00152	-0.04570	0.00000	0.00
8	0.00000	0.00000	0.00000	0.00000	0.99999	0.00436	0.00000	0.00
9	0.00000	0.00000	0.00000	1.00000	0.00028	0.03774	0.00000	0.00

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.12688	0.42640	0.00003	0.00004
2	0.51850	0.04854	1.11989	-0.05298
3	0.46734	-9.35989	-0.00044	0.00008
4	0.00376	-0.00864	0.42183	-0.05875
5	0.39113	0.15286	0.00000	0.00011
6	0.02811	0.00552	0.00887	0.19699
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.3: Linear Model for Airspeed of 60 knots, Climb Rate 0 ft/min.

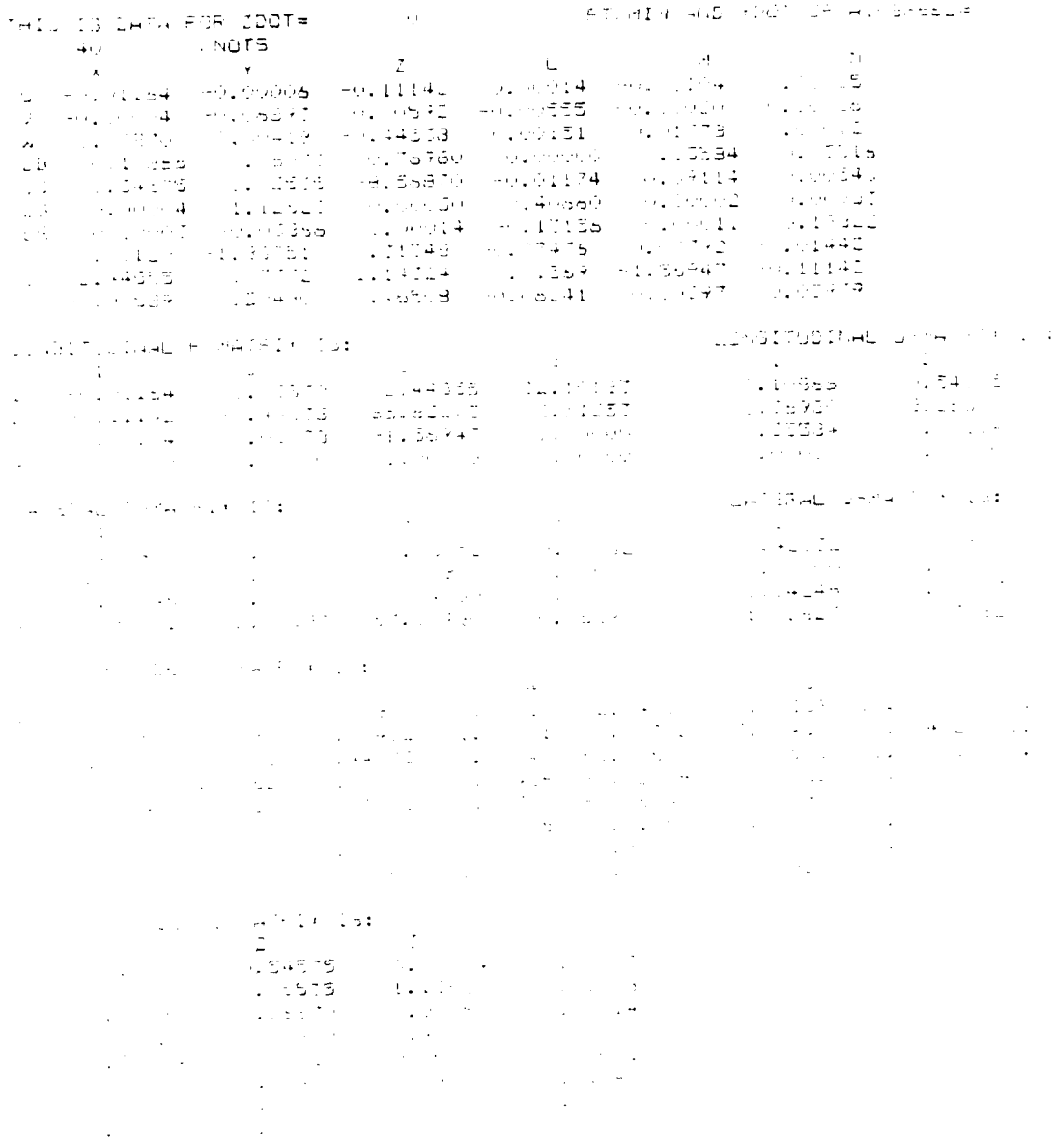


Figure G.4: **Linear Model for Airspeed of 40 knots, Climb Rate 0 ft/min.**

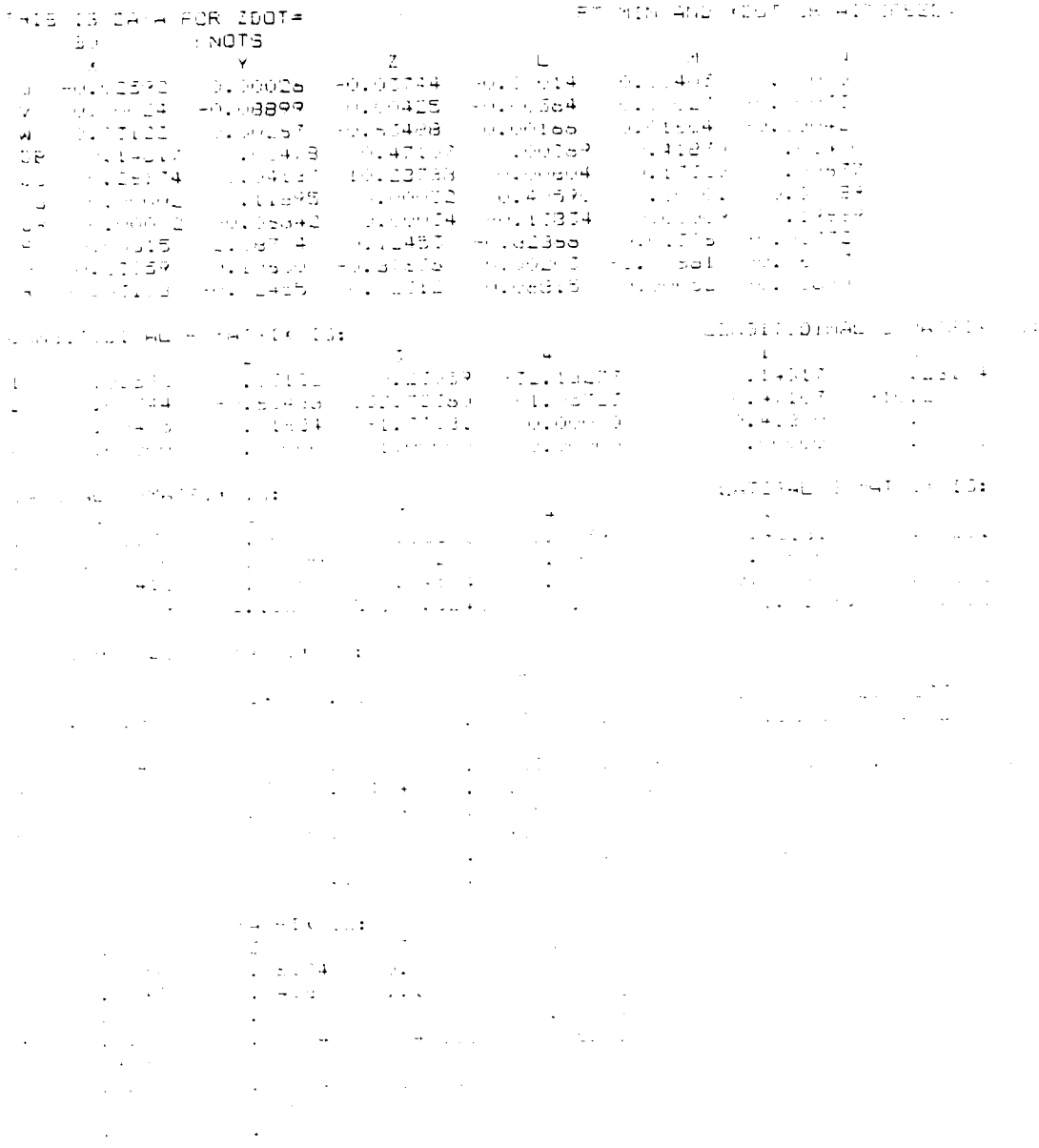


Figure G.5: **Linear Model for Airspeed of 80 knots, Climb Rate 0 ft/min.**

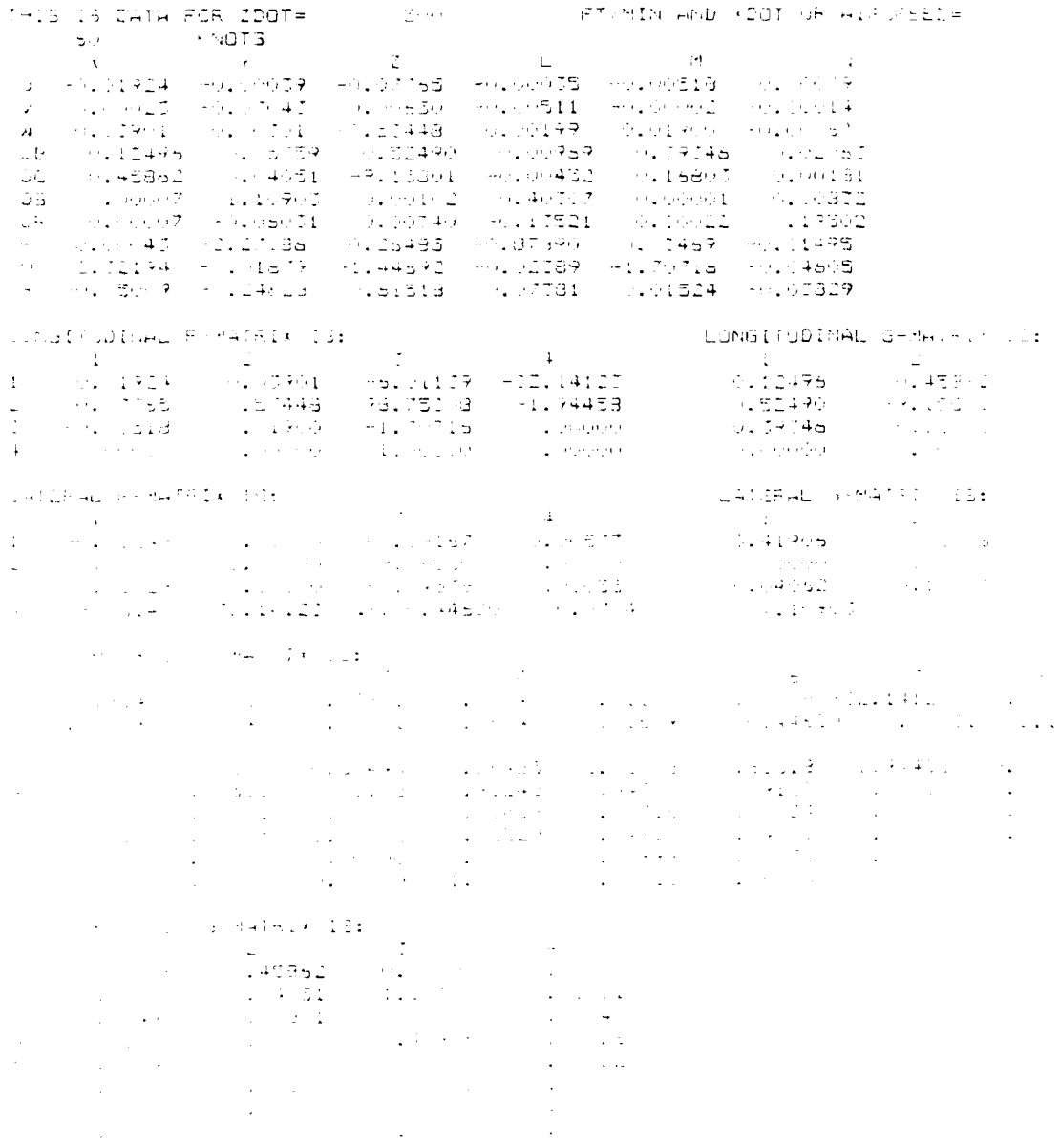


Figure G.6: **Linear Model for Airspeed of 60 knots, Climb Rate 500 ft/min.**

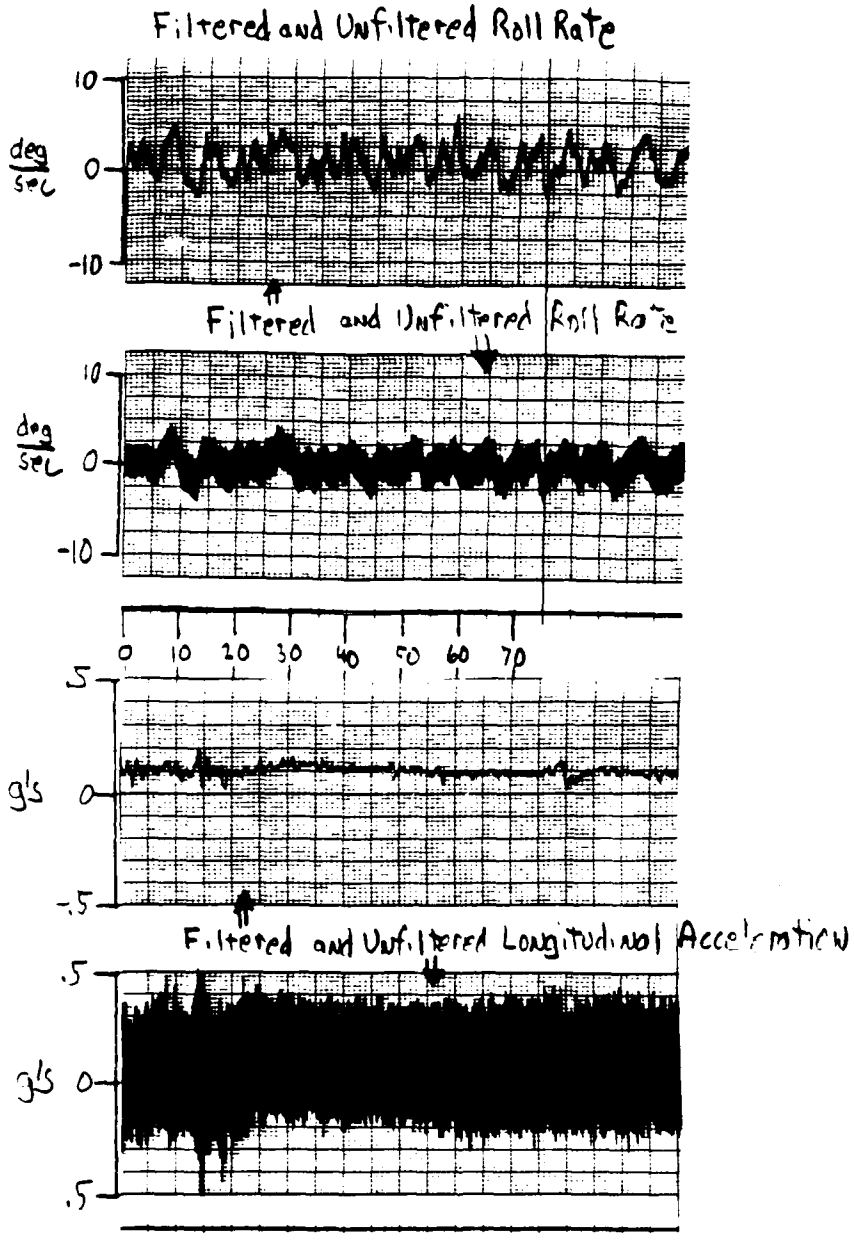


Figure H.2: Comparison of Filtered and Unfiltered Data in Flight. The importance of the filters is evident here where -data- is shown before and after the filter.

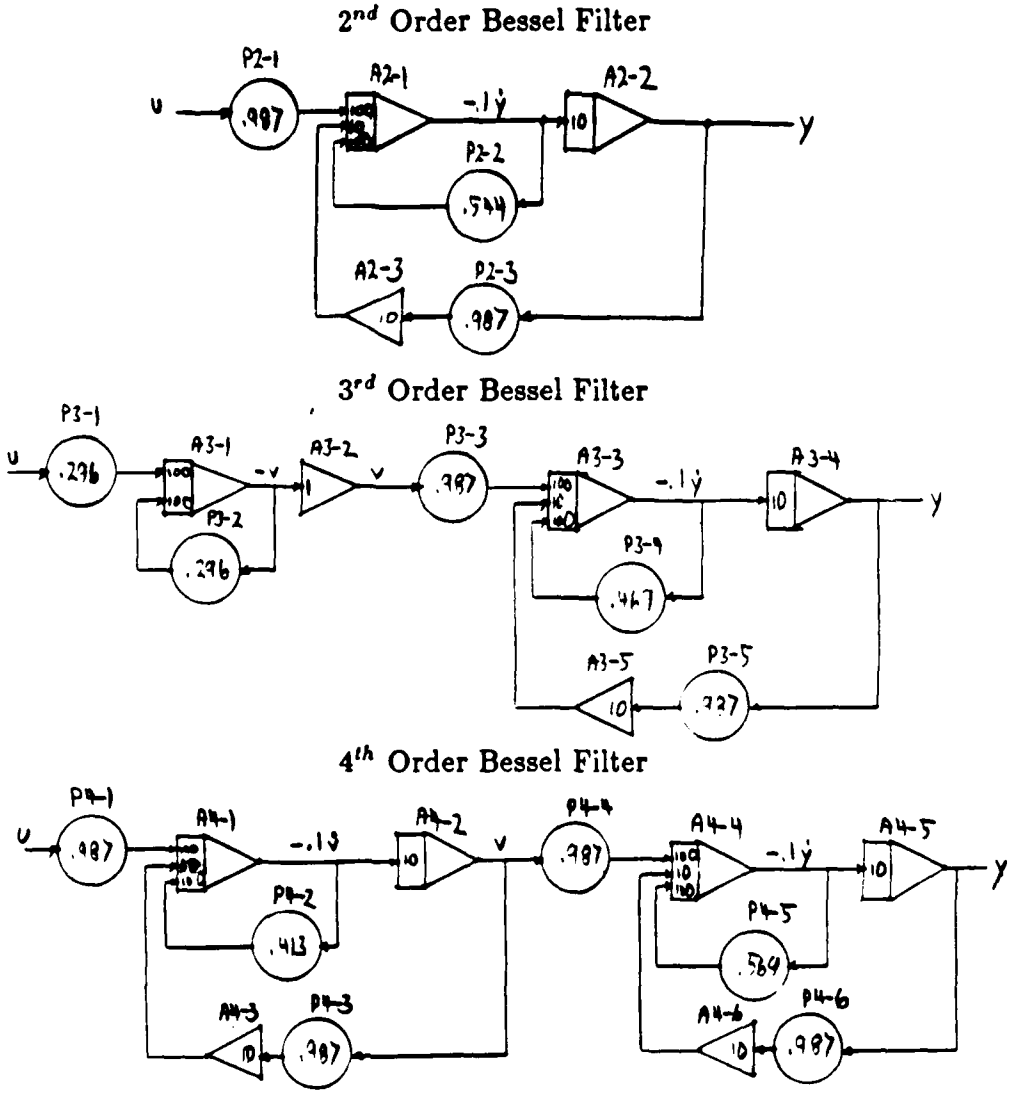


Figure H.1: **Bessel Filter Analog Flow Diagrams.** These filters were programmed on the airborne TR-48 analog computer.

shown in Figure H.2 where the unfiltered and filtered measurements are compared. By the end of flight test, all these filters were replaced by hardwired 3^{rd} order Bessel filters located in a signal conditioning box.

Appendix H.

Bessel Filters

The Bessel filters described here were designed to eliminate the "3 per rev" and "6 per rev" harmonics at 11 Hz and 22 Hz due to the 225 rpm rotor. The break frequency was chosen at 5 Hz as a compromise between noise attenuation and measurement bandwidth. The actual filter designs came from reference 15. Nine filters were patched on the airborne TR-48 analog computer:

4^{th} order– Body axis accelerations (A_x, A_y, A_z)

3^{rd} order– Roll rate (p)

2^{nd} order– Pitch rate (q), Yaw rate (r), Velocity (u), Altitude (h)

The transfer functions for these filters are shown below:

$$\frac{974603}{s^4 + 98s^3 + 4323s^2 + 96906s + 974603}$$

$$\frac{29220}{s^3 + 76s^2 + 2374s + 29220} \qquad (H.1)$$

$$\frac{987}{s^2 + 54s + 987}$$

Figure H.1 shows the analog patch diagrams for these filters. Their effectiveness is

THIS IS DATA FOR ZDOT= -500 FT/MIN AND XDOT OR AIRSPEED=
20 KNOTS

	X	Y	Z	L	M	N
U	-0.00641	0.00087	-0.11816	-0.00012	0.00468	0.00050
V	0.00053	-0.11666	0.00208	-0.00619	-0.00088	0.00135
W	0.03589	0.00354	-0.37819	0.00067	0.00836	0.00028
DB	0.11223	0.03487	0.15524	-0.01678	0.34191	0.04829
DC	0.70594	0.06762	-8.31674	-0.01528	0.03169	0.00386
DS	-0.00007	1.15112	0.00073	0.41279	0.00000	0.01026
DR	-0.00005	-0.03522	-0.00002	-0.13336	-0.00015	0.20260
P	-0.00190	-1.40184	-0.09404	-0.66455	-0.05462	-0.01072
Q	2.52291	-0.02457	0.38001	0.07991	-1.31713	-0.16680
R	-0.02995	-0.15189	-0.44359	-0.04845	-0.00624	-0.04239

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	-0.00641	0.03589	10.85624	-32.05505
2	-0.11816	-0.37819	33.78001	-3.05181
3	0.00468	0.00836	-1.31713	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

	1	2
1	0.11223	0.70594
2	0.15524	-8.31674
3	0.34191	0.03169
4	0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.68970	0.00000	-0.06718	-0.00584
2	1.00000	0.00000	0.09521	0.00000
3	-0.06389	0.00000	-0.04757	0.00090
4	-9.73517	32.05505	-33.55189	-0.11666

LATERAL G-MATRIX IS:

	1	2
1	0.42988	-0.05528
2	0.00000	0.00000
3	0.04340	0.19834
4	1.15112	-0.03522

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	-0.00641	0.00053	0.03589	-0.00190	10.85624	-0.02995	-32.05505	0.00000
2	0.00087	-0.11666	0.00354	-9.73517	-0.02457	-33.55189	0.02449	32.05403
3	-0.11816	0.00208	-0.37819	-0.09404	33.78001	-0.44359	-3.05181	0.00000
4	0.00008	-0.00584	0.00080	-0.68970	0.01469	-0.06718	0.00000	0.00000
5	0.00468	-0.00088	0.00836	-0.05462	-1.31713	-0.00624	0.00000	0.00000
6	0.00051	0.00090	0.00034	-0.06389	-0.16567	-0.04757	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.99997	0.00802	0.00000	0.00000
8	0.00000	0.00000	0.00000	1.00000	0.00076	-0.03520	0.00000	0.00000

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.11223	0.70594	-0.00007	-0.00005
2	0.03487	0.06762	1.15112	-0.03522
3	0.15524	-8.31674	0.00073	-0.00002
4	0.00230	-0.01419	0.42988	-0.05528
5	0.34191	0.03169	0.00000	-0.00015
6	0.04847	-0.00277	0.04340	-0.01060
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.16: **Linear Model for Airspeed of 20 knots, Climb Rate -500 ft/min.**

THIS IS DATA FOR ZDOT= 500 FT/MIN AND XDOT OR AIRSPEED= 20 KNOTS

	X	Y	Z	L	M	N
U	0.00129	0.00122	-0.16248	0.00027	0.00965	0.00031
V	-0.00143	-0.10831	0.01176	-0.00563	0.00097	0.00131
W	0.03219	0.00341	-0.31546	0.00096	0.01226	0.00010
DB	0.10490	0.04843	0.16602	-0.00798	0.32750	0.03966
DC	0.69235	0.05507	-8.15329	-0.01434	0.00333	0.00324
DS	0.00013	1.14249	-0.00136	0.41067	0.00000	0.01006
DR	0.00008	-0.03710	0.00018	-0.13291	0.00027	0.20108
P	0.01602	-1.73729	0.31697	-0.74894	0.08761	-0.00977
Q	2.52136	-0.06817	-0.09012	0.04674	-1.34224	-0.13206
R	-0.17655	-0.19694	1.42350	-0.06014	0.01388	-0.04086

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	0.00129	0.03219	-5.81197	-32.05065
2	-0.16248	-0.31546	33.30988	-3.09780
3	0.00965	0.01226	-1.34224	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

	1	2
	0.10490	0.69235
	0.16602	-8.15329
	0.32750	0.00333
	0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.77635	0.00000	-0.07861	-0.00527
2	1.00000	0.00000	0.09665	0.00000
3	-0.06962	0.00000	-0.04692	0.00090
	6.59604	32.05065	-33.59694	-0.10831

LATERAL G-MATRIX IS:

	1	2
	0.42761	-0.05543
	0.00000	0.00000
	0.04302	0.19681
	1.14249	-0.03710

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	0.00129	-0.00143	0.03219	0.01602	-5.811[illegible]7	-0.17655	-32.05065	[illegible]
2	0.00122	-0.10831	0.00341	6.59604	-0.06817	-33.59694	0.02086	32.04993
3	-0.16248	0.01176	-0.31546	0.31697	33.30988	1.42350	-3.09780	0.[illegible]
4	[illegible]040	-0.00527	0.00103	-0.77635	-0.00541	-0.07361	0.00000	0.[illegible]
5	0.00965	0.00097	0.01226	0.08761	-1.34224	0.01388	0.00000	0.[illegible]
6	[illegible]4	0.00090	0.00018	-0.06962	-0.13249	-0.04692	0.00000	0.[illegible]
7	[illegible]	0.00000	0.00000	0.00000	0.99998	0.00037	0.00000	0.[illegible]
8	[illegible]	0.00000	0.00000	1.00000	0.00065	-0.00065	0.00000	0.[illegible]

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.10490	0.69235	0.00013	0.00008
2	0.04843	0.05507	1.14249	-0.03710
3	0.16602	-8.15329	-0.00136	0.00018
4	0.00787	-0.01347	0.42761	-0.05543
5	0.32750	0.00333	0.00000	0.00027
6	0.04027	0.00220	0.04302	-0.01057
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.15: Linear Model for Airspeed of 20 knots, Climb Rate 500 ft/min.

THIS IS DATA FOR ZDOT= 0 FT/MIN AND XDOT OR AIRSPEED= 20 KNOTS

	X	Y	Z	L	M	N
U	-0.00259	0.00095	-0.14099	0.00004	0.00679	0.00042
V	-0.00046	-0.11155	0.00524	-0.00591	-0.00006	0.00132
W	0.03412	0.00334	-0.34885	0.00077	0.00992	0.00021
DB	0.10849	0.04076	0.16269	-0.01247	0.33528	0.04371
DC	0.69883	0.06166	-8.21477	-0.01453	0.02107	0.00344
DS	0.00004	1.14599	-0.00040	0.41153	0.00000	0.01014
DR	0.00002	-0.03632	0.00012	-0.13309	0.00009	0.20170
P	0.00784	-1.57519	0.18709	-0.70862	0.03618	-0.00921
Q	2.51853	-0.04279	0.21293	0.06396	-1.31598	-0.14872
R	-0.09582	-0.17614	0.39539	-0.05466	-0.00294	-0.04145

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	-0.00259	0.03412	2.51853	-32.05281
2	-0.14099	-0.34885	33.61293	-3.07534
3	0.00679	0.00992	-1.31598	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

	1	2
	0.10849	0.69883
	0.16269	-8.21477
	0.33528	0.02107
	0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.77454	0.00000	-0.07320	-0.00556
2	1.00000	0.00000	0.09595	0.00000
3	-0.06583	0.00000	-0.04709	0.00089
	-1.57519	32.05281	-33.57614	-0.11155

LATERAL G-MATRIX IS:

	1	2
	0.42853	-0.05537
	0.00000	0.00000
	0.04317	0.19743
	1.14599	-0.03632

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	-0.00259	-0.00046	0.03412	0.00784	2.51853	-0.09582	-32.05281	0.00000
2	0.00095	-0.11155	0.00334	-1.57519	-0.04279	-33.57614	0.02265	32.05194
3	-0.14099	0.00524	-0.34885	0.18709	33.61293	0.39539	-3.07534	0.00000
4	0.00021	-0.00556	0.00088	-0.77454	0.00558	-0.07320	0.00000	0.00000
5	0.00679	-0.00006	0.00992	0.03618	-1.31598	-0.00294	0.00000	0.00000
6	0.00044	0.00089	0.00028	-0.06583	-0.14829	-0.04709	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.99997	0.00776	0.00000	0.00000
8	0.00000	0.00000	0.00000	1.00000	0.00071	0.09594	0.00000	0.00000

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.10849	0.69883	0.00004	0.00002
2	0.04076	0.06166	1.14599	-0.03632
3	0.16269	-8.21477	-0.00040	0.00012
4	0.00489	-0.01359	0.42853	-0.05537
5	0.33528	0.02107	0.00000	0.00009
6	0.04409	-0.00279	0.04317	-0.01058
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.14: Linear Model for Airspeed of 20 knots, Climb Rate 0 ft/min.

THIS IS DATA FOR ZDOT= -500 FT/MIN AND XDOT OR AIRSPEED= 0 KNOTS

	X	Y	Z	L	M	N
U	-0.02325	-0.00038	0.02826	-0.00037	0.00623	0.00050
V	-0.00097	-0.14525	0.00135	-0.00635	0.00008	-0.00070
W	0.03589	0.00301	-0.32741	0.00047	0.00203	0.00026
DB	0.11831	0.01249	0.02908	-0.02638	0.33695	0.05075
DC	0.94701	0.06887	-8.13578	-0.01448	0.02009	0.00098
DS	-0.00015	1.16274	0.00125	0.41643	0.00000	0.00924
DR	-0.00008	-0.05288	-0.00009	-0.13909	-0.00026	0.20448
P	0.03242	-1.32917	-0.17822	-0.65429	0.00108	0.00184
Q	2.59202	0.00144	0.46157	0.10871	-1.24188	-0.17600
R	-0.12040	-0.14595	0.31169	-0.04691	-0.00288	-0.04263

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	-0.02325	0.03589	10.92535	-31.98588
2	0.02826	-0.32741	0.46157	-3.70723
3	0.00623	0.00203	-1.24188	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

	1	2
1	0.11831	0.94701
2	0.02908	-8.13578
3	0.33695	0.02009
4	0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.67402	0.00000	-0.06569	-0.00683
2	1.00000	0.00000	0.11590	0.00000
3	-0.05012	0.00000	-0.04769	-0.00123
4	-9.66250	31.98588	-0.14595	-0.14525

LATERAL G-MATRIX IS:

	1	2
1	0.43322	-0.06042
2	0.00000	0.00000
3	0.04264	0.19982
4	1.16274	-0.05288

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	-0.02325	-0.00097	0.03589	0.03242	10.92535	-0.12040	-31.98588	0.00000
2	-0.00038	-0.14525	0.00301	-9.66250	0.00144	-0.14595	0.03116	31.98475
3	0.02826	0.00135	-0.32741	-0.17822	0.46157	0.31169	-3.70723	0.00000
4	-0.00018	-0.00683	0.00059	-0.67402	0.04066	-0.06569	0.00000	0.00000
5	0.00623	0.00008	0.00203	0.00108	-1.24188	-0.00288	0.00000	0.00000
6	0.00049	-0.00123	0.00031	-0.05012	-0.17287	-0.04769	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.99995	0.00846	0.00000	0.00000
8	0.00000	0.00000	0.00000	1.00000	0.00097	0.11590	0.00000	0.00000

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.11831	0.94701	-0.00015	-0.00008
2	0.01249	0.06887	1.16274	-0.05288
3	0.02908	-8.13578	0.00125	-0.00009
4	-0.00660	-0.01454	0.43322	-0.06042
5	0.33695	0.02009	0.00000	-0.00026
6	0.05024	-0.00014	0.04264	-0.01106
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.13: Linear Model for Airspeed of 0 knots, Climb Rate -500 ft/min.

THIS IS DATA FOR ZDOT= 500 FT/MIN AND XDOT OR AIRSPEED= 0 KNOTS

	X	Y	Z	L	M	N
U	-0.01857	-0.00001	0.02097	-0.00014	0.01356	0.00023
V	-0.00127	-0.13179	0.00975	-0.00583	0.00027	-0.00055
W	0.02894	0.00244	-0.25982	0.00052	0.00285	0.00018
DB	0.10910	0.01113	0.03148	-0.02213	0.31958	0.04294
DC	0.93710	0.05696	-8.05096	-0.01419	0.01764	0.00052
DS	0.00012	1.15717	-0.00110	0.41509	0.00000	0.00906
DR	0.00007	-0.05506	0.00005	-0.13905	0.00021	0.20349
P	-0.02869	-1.64496	0.35800	-0.73128	0.01003	-0.00148
Q	2.57072	0.00500	0.40010	0.09244	-1.24855	-0.14708
R	-0.08620	-0.18117	0.43277	-0.05704	0.01044	-0.04078

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	-0.01857	0.02894	-5.76261	-31.98550
2	0.02097	-0.25982	0.40010	-3.71050
3	0.01356	0.00285	-1.24855	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

	1	2
	0.10910	0.93710
	0.03148	-8.05096
	0.31958	0.01764
	0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.75477	0.00000	-0.07538	-0.00624
2	1.00000	0.00000	0.11601	0.00000
3	-0.05966	0.00000	-0.04659	-0.00103
	6.68837	31.98550	-0.18117	-0.13179

LATERAL G-MATRIX IS:

	1	2
	0.43176	-0.06079
	0.00000	0.00000
	0.04234	0.19880
	1.15717	-0.05506

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	-0.01857	-0.00127	0.02894	-0.02869	-5.76261	-0.08620	-31.98550	0.00000
2	-0.00001	-0.13179	0.00244	6.68837	0.00500	-0.18117	0.02737	31.98464
3	0.02097	0.00975	-0.25982	0.35800	0.40010	0.43277	-3.71050	0.00000
4	-0.00005	-0.00624	0.00061	-0.75477	0.03562	-0.07538	0.00000	0.00000
5	0.01356	0.00027	0.00285	0.01003	-1.24855	0.01044	0.00000	0.00000
6	0.00023	-0.00103	0.00023	-0.05966	-0.14433	-0.04659	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.99997	0.00738	0.00000	0.00000
8	0.00000	0.00000	0.00000	1.00000	0.00086	0.11600	0.00000	0.00000

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.10910	0.93710	0.00012	0.00007
2	0.01113	0.05696	1.15717	-0.05506
3	0.03148	-8.05096	-0.00110	0.00005
4	-0.00539	-0.01442	0.43176	-0.06079
5	0.31958	0.01764	0.00000	0.00021
6	0.04252	-0.00059	0.04234	-0.01105
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.12: Linear Model for Airspeed of 0 knots, Climb Rate 500 ft/min.

THIS IS DATA FOR ZDOT= 0 FT/MIN AND XDOT OR AIRSPEED= 0 KNOTS

	X	Y	Z	L	M	N
U	-0.02114	-0.00019	0.02484	-0.00028	0.00925	0.00040
V	-0.00085	-0.13712	0.00374	-0.00608	0.00017	-0.00062
W	0.03259	0.00265	-0.29557	0.00048	0.00234	0.00022
DB	0.11408	0.01175	0.30310	-0.02410	0.32921	0.04655
DC	0.93867	0.06253	-8.06188	-0.01403	0.01905	0.00072
DS	0.00000	1.15902	-0.00002	0.41552	0.00000	0.00914
DR	-0.00001	-0.05395	0.00011	-0.13896	0.00000	0.20382
P	0.02051	1.49383	0.04190	-0.69495	0.04267	0.00080
Q	2.58521	0.00414	0.43507	0.10020	-1.22925	-0.16052
R	-0.10552	-0.16450	0.36222	-0.05220	-0.00433	-0.04172

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	-0.02114	0.03259	2.58521	-31.98577
2	0.02484	-0.29557	0.43507	-3.70815
3	0.00925	0.00234	-1.22925	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

	1	2
1	0.11408	0.93867
2	0.30310	-8.06188
3	0.32921	0.01905
4	0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.71638	0.00000	-0.07077	-0.00652
2	1.00000	0.00000	0.11593	0.00000
3	-0.05442	0.00000	-0.04718	-0.00112
4	-1.49383	31.98577	-0.16450	-0.13712

LATERAL G-MATRIX IS:

	1	2
1	0.43224	-0.06056
2	0.00000	0.00000
3	0.04246	0.19915
4	1.15902	-0.05395

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	-0.02114	-0.00085	0.03259	0.02051	2.58521	-0.10552	-31.98577	0.00000
2	-0.00019	-0.13712	0.00265	-1.49383	0.00414	-0.16450	0.02922	31.98478
3	0.02484	0.00374	-0.29557	0.04190	0.43507	0.36222	-3.70815	0.00000
4	-0.00013	-0.00652	0.00058	-0.71638	0.07817	-0.07077	0.00000	0.00000
5	0.00925	0.00017	0.00234	0.04267	-1.22925	-0.00433	0.00000	0.00000
6	0.00039	-0.00112	0.00027	-0.05442	-0.15758	-0.04718	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.99997	0.00788	0.00000	0.00000
8	0.00000	0.00000	0.00000	1.00000	0.00091	0.11593	0.00000	0.00000

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.11408	0.93867	0.00000	-0.00001
2	0.01175	0.06253	1.15902	-0.05395
3	0.30310	-8.06188	-0.00002	0.00011
4	-0.00596	-0.01418	0.43224	-0.06056
5	0.32921	0.01905	0.00000	0.00000
6	0.04609	-0.00027	0.04246	-0.01105
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.11: Linear Model for Airspeed of 0 knots, Climb Rate 0 ft/min.

THIS IS DATA FOR ZDOT= -500 FT/MIN AND XDOT OR AIRSPEED=
-20 KNOTS

	X	Y	Z	L	M	N
U	-0.04431	-0.00170	0.18208	-0.00058	0.00498	0.00033
V	-0.00184	-0.14266	-0.00104	-0.00631	0.00121	-0.00271
W	0.04284	0.00272	-0.34017	0.00049	-0.00447	0.00025
DB	0.13510	-0.01042	-0.09822	-0.03337	0.34340	0.04849
DC	1.22694	0.06576	-8.22802	-0.01187	0.01087	-0.00194
DS	-0.00025	1.16691	0.00182	0.41832	0.00000	0.00791
DR	-0.00013	-0.07451	-0.00001	-0.14503	-0.00037	0.20502
P	0.07629	-1.41972	-0.31843	-0.68419	0.06421	0.01341
Q	2.54761	0.02677	0.57144	0.12586	-1.31840	-0.16553
R	-0.21772	-0.15751	1.13901	-0.05023	-0.00735	-0.04199

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	-0.04431	0.04284	10.88094	-31.90117
2	0.18208	-0.34017	-32.82856	-4.37670
3	0.00498	-0.00447	-1.31840	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

	1	2
1	0.13510	1.22694
2	-0.09822	-8.22802
3	0.34340	0.01087
4	0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.70016	0.00000	-0.06885	-0.00761
2	1.00000	0.00000	0.13729	0.00000
3	-0.04049	0.00000	-0.04729	-0.00330
	-9.75305	31.90117	33.24249	-0.14266

LATERAL G-MATRIX IS:

	1	2
1	0.43462	-0.06633
2	0.00000	0.00000
3	0.04137	0.13992
	1.16691	-0.07451

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	-0.04431	-0.00184	0.04284	0.07629	10.88094	-0.21772	-31.90117	0.00000
2	-0.00170	-0.14266	0.00272	-9.75305	0.02677	33.24249	0.00467	31.90117
3	0.18208	-0.00104	-0.34017	-0.31843	-32.82856	1.13901	-4.37670	0.00000
4	-0.00046	-0.00761	0.00061	-0.70016	0.06259	-0.06885	0.00000	0.00000
5	0.00498	0.00121	-0.00447	0.06421	-1.31840	-0.00735	0.00000	0.00000
6	0.00029	-0.00330	0.00030	-0.04049	-0.18072	-0.04729	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.99997	0.00792	0.00000	0.00000
8	0.00000	0.00000	0.00000	1.00000	0.00109	0.13719	0.00000	0.00000

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.13510	1.22694	-0.00025	-0.00013
2	-0.01042	0.06576	1.16691	-0.07451
3	-0.09822	-8.22802	0.00182	-0.00001
4	-0.01473	-0.01303	0.43462	-0.06633
5	0.34340	0.01087	0.00000	-0.00037
6	0.04736	-0.00294	0.04137	-0.01152
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.10: Linear Model for Airspeed of -20 knots, Climb Rate -500 ft/min.

THIS IS DATA FOR ZDOT= 500 FT/MIN AND XDOT OR AIRSPEED= -20 KNOTS

	X	Y	Z	L	M	N
U	-0.04422	-0.00151	0.21206	-0.00063	0.01045	0.00008
V	-0.00058	-0.12523	0.00570	-0.00574	-0.00033	-0.00242
W	0.03283	0.00221	-0.27064	0.00530	-0.00731	0.00030
DB	0.12940	-0.02766	-0.10331	-0.03274	0.32931	0.03949
DC	1.20437	0.05345	-8.06621	-0.01071	0.03615	-0.00256
DS	0.00007	1.15921	-0.00050	0.41645	0.00001	0.00772
DR	0.00003	-0.07663	-0.00001	-0.14475	0.00009	0.20366
P	-0.05257	-1.76946	0.30099	-0.76640	-0.05053	0.00403
Q	2.46181	0.08166	0.93621	0.12011	-1.33744	-0.12984
R	0.00776	-0.18787	-0.45957	-0.06008	0.00643	-0.03976

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	-0.04422	0.03283	-5.87152	-31.90770
2	0.21206	-0.27064	-32.46379	-4.32879
3	0.01045	-0.00731	-1.33744	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

	1	2
	0.12940	1.20437
	-0.10331	-8.06621
	0.32931	0.03615
	0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.78875	0.00000	-0.07810	-0.00690
2	1.00000	0.00000	0.13567	0.00000
3	-0.05677	0.00000	-0.04578	-0.00295
	6.56387	31.90770	33.21213	-0.12523

LATERAL G-MATRIX IS:

	1	2
	0.43262	-0.06659
	0.00000	0.00000
	0.04107	0.19853
	1.15921	-0.07663

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	-0.04422	-0.00058	0.03283	-0.05257	-5.87152	0.00776	-31.90770	0.00000
2	-0.00151	-0.12523	0.00221	6.56387	0.08166	33.21213	0.02901	31.90699
3	0.21206	0.00570	-0.27064	0.30099	-32.46379	-0.45957	-4.32879	0.00000
4	-0.00062	-0.00690	0.00559	-0.78875	0.07115	-0.07810	0.00000	0.00000
5	0.01045	-0.00033	-0.00731	-0.05053	-1.33744	0.00643	0.00000	0.00000
6	0.00003	-0.00295	0.00073	-0.05677	-0.12436	-0.04578	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.99998	0.00670	0.00000	0.00000
8	0.00000	0.00000	0.00000	1.00000	0.00091	0.13566	0.00000	0.00000

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.12940	1.20437	0.00007	0.00003
2	-0.02766	0.05345	1.15921	-0.07663
3	-0.10331	-8.06621	-0.00050	-0.00001
4	-0.01773	-0.01208	0.43262	-0.06659
5	0.32931	0.03615	0.00001	0.00009
6	0.03812	-0.00249	0.04107	-0.01151
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.9: **Linear Model for Airspeed of -20 knots, Climb Rate 500 ft/min.**

THIS IS DATA FOR ZDOT= 0 FT/MIN AND XDOT OR AIRSPEED= -20 KNOTS

	X	Y	Z	L	M	N
U	-0.04449	-0.00149	0.19817	-0.00059	0.00727	0.00023
V	-0.00075	-0.13255	0.00100	-0.00601	0.00054	-0.00254
W	0.03811	0.00238	-0.30703	0.00050	-0.00562	0.00026
DB	0.13262	-0.01824	-0.10291	-0.03269	0.33689	0.04375
DC	1.21170	0.05992	-8.12808	-0.01116	0.02013	-0.00218
DS	-0.00007	1.16217	0.00051	0.41715	0.00000	0.00782
DR	-0.00006	-0.07540	0.00014	-0.14475	-0.00012	0.20419
P	0.05290	-1.60055	-0.19944	-0.72706	0.05658	0.00866
Q	2.51458	0.05194	0.69186	0.12183	-1.31503	-0.14697
R	-0.12437	-0.17280	0.47219	-0.05530	-0.01058	-0.04088

LONGITUDINAL F-MATRIX IS:

	1	2	3	4
1	-0.04449	0.03811	2.51458	-31.90471
2	0.19817	-0.30703	-32.70814	-4.35079
3	0.00727	-0.00562	-1.31503	0.00000
4	0.00000	0.00000	1.00000	0.00000

LONGITUDINAL G-MATRIX IS:

1	2
0.13262	1.21170
-0.10291	-8.12808
0.33689	0.02013
0.00000	0.00000

LATERAL F-MATRIX IS:

	1	2	3	4
1	-0.74630	0.00000	-0.07363	-0.00723
2	1.00000	0.00000	0.13637	0.00000
3	-0.04887	0.00000	-0.04656	-0.00310
4	-1.60055	31.90471	33.22720	-0.13255

LATERAL G-MATRIX IS:

1	2
0.43338	-0.06638
0.00000	0.00000
0.04123	0.19907
1.16217	-0.07540

THE 8TH ORDER F-MATRIX IS:

	1	2	3	4	5	6	7	8
1	-0.04449	-0.00075	0.03811	0.05290	2.51458	-0.12437	-31.90471	0.00000
2	-0.00149	-0.13255	0.00238	-1.60055	0.05194	33.22720	0.03181	31.90386
3	0.19817	0.00100	-0.30703	-0.19944	-32.70814	0.47219	-4.35079	0.00000
4	-0.00052	-0.00723	0.00062	-0.74630	0.06597	-0.07363	0.00000	0.00000
5	0.00727	0.00054	-0.00562	0.05658	-1.31503	-0.01058	0.00000	0.00000
6	0.00019	-0.00310	0.00031	-0.04887	-0.14188	-0.04656	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.99997	0.00731	0.00000	0.00000
8	0.00000	0.00000	0.00000	1.00000	0.00100	0.13636	0.00000	0.00000

THE 8TH ORDER G-MATRIX IS:

	1	2	3	4
1	0.13262	1.21170	-0.00007	-0.00006
2	-0.01824	0.05992	1.16217	-0.07540
3	-0.10291	-8.12808	0.00051	0.00014
4	-0.01595	-0.01239	0.43338	-0.06638
5	0.33689	0.02013	0.00000	-0.00012
6	0.04252	-0.00314	0.04123	-0.01151
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000

Figure G.8: **Linear Model for Airspeed of -20 knots, Climb Rate 0 ft/min.**

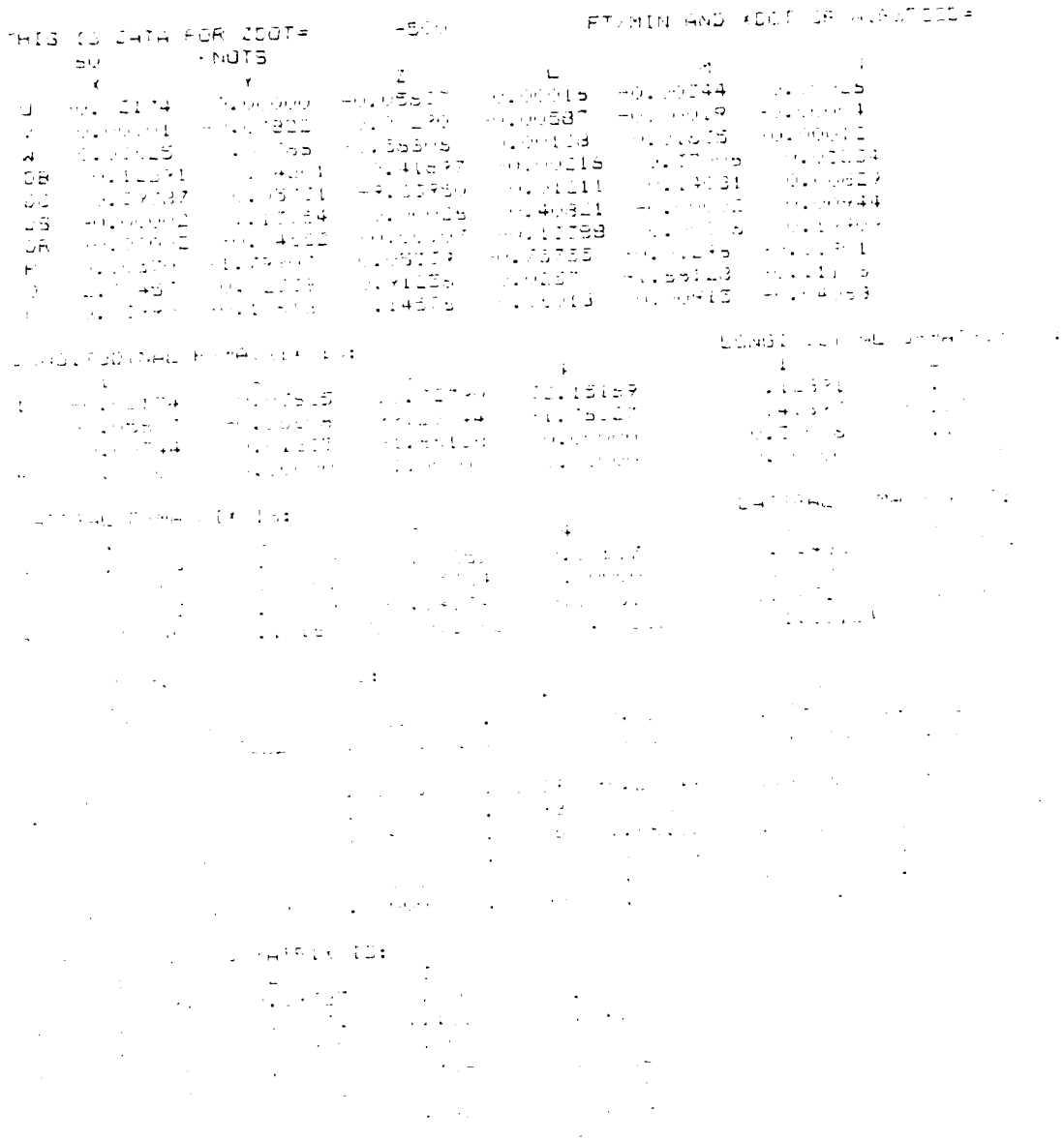


Figure G.7: **Linear Model for Airspeed of 60 knots, Climb Rate -500 ft/min.**

Appendix I.

Flight Software

This appendix lists the flight computer software unique to this project. It is written in Sperry 1819A assembly code and internally documented. Three different sets of the code are shown. The initialization code was executed whenever the experimental control system was disengaged. It did such things as:

- reset the compensator states to zero
- set the trim values of controls and states to the current values of these variables in preparation for system engage

The experimental controller subroutines computed the control laws, described in Chapters 3 and 4, and were executed at 20 Hz. Both the longitudinal CAS and hover controller are shown. The last item shown in this appendix are the instructions which reserve memory for the variables and contants unique to the subroutines.

Initialisation Code.

```
1819A ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION  1                                   PAGE  32

                        *
                        * ****************************************************************
                        *
                        *     MODIFICATIONS TO INIT DONE ON 6 JUN 84
                        *
                        *
                        *       THESE LINES STORE INITIAL THETA AND AIRSPEED FOR THE CMSTR SUBR.
                        *       AND INITIALIZE COMPENSATOR INTEGRATORS.
                        *
002600  120746              ENTAL   IWETH                    GET THETA AT ENGAGE
002601  442156              STRAL   MIRPTH                   STORE THETA AT ENGAGE
002602  442344              STRAL   NIRPTH                   STORE THETA AT ENGAGE
002603  442603              STRAL   OIRPTH                   STORE THETA AT ENGAGE
002604  443151              STRAL   PIRPTH                   STORE THETA AT ENGAGE
002605  120753              ENTAL   INSPHI                   GET PHI AT ENGAGE
002606  442604              STRAL   OIRPPH                   STORE PHI AT ENGAGE
002607  443152              STRAL   PIRPPH                   STORE PHI AT ENGAGE
002610  120752              ENTAL   AIRSPD                   GET THE AIRSPEED AT ENGAGE
002611  442157              STRAL   MIRPAS                   STORE AIRSPEED AT ENGAGE
002612  442345              STRAL   NIRPAS                   STORE AIRSPEED AT ENGAGE
002613  442605              STRAL   OIRPAS                   STORE AIRSPEED AT ENGAGE
002614  443153              STRAL   PIRPAS                   STORE AIRSPEED AT ENGAGE
002615  120765              ENTAL   IVSI                     GET VERT VELOCITY AT ENGAGE
002616  442160              STRAL   MIRPHD                   STORE VERTICAL VELOCITY AT ENGAGE
002617  442346              STRAL   NIRPHD                   STORE VERTICAL VELOCITY AT ENGAGE
002620  442606              STRAL   OIRPHD                   STORE VERTICAL VELOCITY AT ENGAGE
002621  443154              STRAL   PIRPHD                   STORE VERTICAL VELOCITY AT ENGAGE
002622  121134              ENTAL   XPILT                    GET RUNWAY X POSITION
002623  442662              STRAL   OXREF                    CXREF=INITIAL RUNWAY X
002624  443230              STRAL   PXREF                    PXREF=INITIAL RUNWAY X
002625  121147              ENTAL   XBODY                    GET BODY X POSITION
002626  442003              STRAL   OXBREF                   CXBREF=INITIAL BODY X
002627  443231              STRAL   PXBREF                   PXBREF=INITIAL BODY X
002630  121135              ENTAL   YPILT                    GET RUNWAY Y POSITION
002631  442673              STRAL   OYREF                    CYREF=INITIAL Y POSITION
002632  443241              STRAL   PYREF                    PYREF=INITIAL Y POSITION
002633  121150              ENTAL   YBODY                    GET BODY Y POSITION
002634  442674              STRAL   OYBREF                   CYBREF=INITIAL BODY Y
002635  443242              STRAL   PYBREF                   PYBREF=INITIAL BODY Y
002636  121136              ENTAL   ZPILT                    GET Z POSITION
002637  442704              STRAL   OZREF                    CZREF=Z POSITION
002640  443252              STRAL   PZREF                    PZREF=Z POSITION
002641  402712              STRZ    OXDI                     CXDI=0
002642  403200              STRZ    PXDI                     PXDI=0
002643  402714              STRZ    OYDI                     CYDI=0
002644  403202              STRZ    PYDI                     PYDI=0
002645  402716              STRZ    OZDI                     CZDI=0
002646  403204              STRZ    PZDI                     PZDI=0
002647  402713              STRZ    OXDI+1                   CXDI+1=0
002650  403201              STRZ    PXDI+1                   PXDI+1=0
002651  402715              STRZ    OYDI+1                   CYDI+1=0
002652  403203              STRZ    PYDI+1                   PYDI+1=0
002653  402717              STRZ    OZDI+1                   CZDI+1=0
002654  403205              STRZ    PZDI+1                   PZDI+1=0
002655  402672              STRZ    OXDOLD                   CXDOLD=0
002656  403240              STRZ    PXDOLD                   PXDOLD=0
002657  402703              STRZ    OYDOLD                   CYDOLD=0
```

Initialisation Code. (contd)

```
1819A ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION   1                                   PAGE   33

       002660  403251            STRZ     PYDCLD                PYDCLC=0
       002661  402711            STRZ     OZDCLD                CZDCLC=0
       002662  403257            STRZ     PZDCLD                PZDCLC=0
       002663  402671            STRZ     OXDPID                CXDPIC=0
       002664  403237            STRZ     PXDPID                PXDPIC=0
       002665  402702            STRZ     OYDPID                CYDPIC=0
       002666  403250            STRZ     PYDPID                PYDPIC=0
       002667  402710            STRZ     OZDPID                CZDPIC=0
       002670  403256            STRZ     PZDPID                PZDPIC=0
       002671  402720            STRZ     OXEZCN                CXEZON=0 START PID FADE-IN AT ZERC
       002672  402721            STRZ     OYEZCN                CYEZON=0 START PID FADE-IN AT ZERC
       002673  402722            STRZ     OZEZCN                CZEZON=0 START PID FADE-IN AT ZERC
       002674  403266            STRZ     PXEZCN                PXEZON=0 START PID FADE-IN AT ZERC
       002675  403267            STRZ     PYEZCN                PYEZON=C START PID FADE-IN AT ZERC
       002676  403270            STRZ     PZEZCN                PZEZON=0 START PID FADE-IN AT ZERC
                        *   INITIALIZATIONS OF THE BRANCHING FLAGS SINCE ESK USES CURRENT
                        *   MEMORY BANK, I.E. BANK 0.
       002677  122145            ENTAL    MNC                   AL:=(MNC) NUMBER OF CONTROLS, CMSTR1
       002700  102144            ENTAU    MR                    AU:=(MR) NUMBER OF COMP STATES, CPSTR1
       002701  507310            ENTSR    10                    USE BANK 0 LOCATIONS
       002702  717776            ADDALK   -1                    AL:=(AL)-1=MC-1
       002703  444457            STRAL    MNCP1                 MNCM1:=MC-1
       002704  507500            XFRUL                          AL:=AU
       002705  717776            ADDALK   -1                    AL:=AL-1=MR-1
       002706  444454            STRAL    MRRP1                 MRRM1:=MR-1
       002707  507313            ENTSR    13                    USE BANK 3 AGAIN
       002710  122333            ENTAL    NNC                   AL:=(NNC) NUMBER OF CONTROLS, CMSTR2
       002711  102332            ENTAU    NRCATA                AU:=(NRCATA) NUMBER OF COMP STATES, CMSTR2
       002712  507310            ENTSR    10                    USE BANK 0 LOCATIONS
       002713  717776            ADDALK   -1                    AL:=(AL)-1=NC-1
       002714  445170            STRAL    NNCP1                 NNCM1:=NC-1
       002715  507500            XFRUL                          AL:=AU
       002716  717776            ADDALK   -1                    AL:=AL-1=NR-1
       002717  445165            STRAL    WRMP1                 NRRM1:=NR-1
       002720  507313            ENTSR    13                    USE BANK 3 AGAIN
       002721  122570            ENTAL    ONC                   AL:=(ONC) NUMBER OF CONTROLS, CPSTR3
       002722  102567            ENTAU    OR                    AU:=(CR) NUMBER OF COMP STATES, CPSTR3
       002723  507310            ENTSR    10                    USE BANK 0 LOCATIONS
       002724  717776            ADDALK   -1                    AL:=(AL)-1=NC-1
       002725  445701            STRAL    ONCP1                 CNCM1:=NC-1
       002726  507500            XFRUL                          AL:=AU
       002727  717776            ADDALK   -1                    AL:=AL-1=CR-1
       002730  445676            STRAL    ORRP1                 CRRM1:=KR-1
       002731  507313            ENTSR    13                    USE BANK 3 AGAIN
       002732  123136            ENTAL    PNC                   AL:=(PNC) NUMBER OF CONTROLS, CMSTR4
       002733  103135            ENTAU    PR                    AU:=(PR) NUMBER OF COMP STATES, CPSTR4
       002734  507310            ENTSR    10                    USE BANK 0 LOCATIONS
       002735  717776            ADDALK   -1                    AL:=(AL)-1=NC-1
       002736  446715            STRAL    PNCP1                 PNCM1:=NC-1
       002737  507500            XFRUL                          AL:=AU
       002740  717776            ADDALK   -1                    AL:=AL-1=NR-1
       002741  446712            STRAL    PRRP1                 PRRM1:=PR-1
       002742  507313            ENTSR    13                    USE BANK 3 AGAIN
                        *   DO THE CMSTR1 INITIALIZATIONS
       002743  507201            ENTICR   1                     USE ICR1 AS INDEX REGISTER
       002744  360000            ENTBK    0                     ICR1:=0
       002745  412034   MINIT1   STRZB    MUI                   (MUI+ICR1):=0
```

Initialisation Code. (contd)

```
1819A ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION   1                                              PAGE   34

002746  564457            ESK     MACP1                                    IF P=MC-1 THEN SKIP A LINE
                 $                        ELSE INITIALIZE THE NEXT MUI AT ZERC, ICR1=ICR1+1
002747  342745            JP      MINIT1                                   CONTINUE MUI INITIALIZATION
002750  360000            ENTBK   0                                        ICR1:=0  START MZK,MZKP1 INITIALIZATION
002751  412114   MINIT2   STRZB   MZK                                      (MZK+ICR1):=0
002752  412116            STRZB   MZKP1                                    (MZKP1+ICR1):=0
002753  564454            ESK     MRRP1                                    IF B=MR-1 THEN SKIP A LINE
                 $                        ELSE GO INITIALIZE NEXT TERM, ICR1=ICR1+1
002754  342751            JP      MINIT2                                   CONTINUE INITIALIZATION
                 $    NOW DO THE CMSTR2 INITIALIZATIONS
002755  507201            ENTICR  1                                        USE ICR1 AS INDEX REGISTER
002756  360000            ENTBK   0                                        ICR1:=0
002757  412211   NINIT1   STRZB   NUI                                      (NUI+ICR1):=0
002760  565170            ESK     NNCP1                                    IF P=NC-1 THEN SKIP A LINE
                 $                        ELSE INITIALIZE THE NEXT NUI AT ZERC, ICR1=ICR1+1
002761  342757            JP      NINIT1                                   CONTINUE NUI INITIALIZATION
002762  360000            ENTBK   0                                        ICR1:=0  START NZK,NZKP1 INITIALIZATION
002763  412271   NINIT2   STRZB   NZK                                      (NZK+ICR1):=0
002764  412274            STRZB   NZKP1                                    (NZKP1+ICR1):=0
002765  565165            ESK     NRRP1                                    IF B=NR-1 THEN SKIP A LINE
                 $ .                      ELSE GO INITIALIZE NEXT TERM, ICR1=ICR1+1
002766  342763            JP      NINIT2                                   CONTINUE INITIALIZATION
                 $    DO THE CMSTR3 INITIALIZATIONS
002767  507201            ENTICR  1                                        USE ICR1 AS INDEX REGISTER
002770  360000            ENTBK   0                                        ICR1:=0  START OZK,OZKP1 INITIALIZATION
002771  412451   OINIT2   STRZB   OZK                                      (OZK+ICR1):=0
002772  412456            STRZB   OZKP1                                    (OZKP1+ICR1):=0
002773  565676            ESK     ORRP1                                    IF B=OR-1 THEN SKIP A LINE
                 $                        ELSE GO INITIALIZE NEXT TERM, ICR1=ICR1+1
002774  342771            JP      OINIT2                                   CONTINUE INITIALIZATION
                 $    NOW DO THE CMSTR4 INITIALIZATIONS
002775  507201            ENTICR  1                                        USE ICR1 AS INDEX REGISTER
002776  360000            ENTBK   0                                        ICR1:=0  START PZK,PZKP1 INITIALIZATION
002777  413017   PINIT2   STRZB   PZK                                      (PZK+ICR1):=0
003000  413024            STRZB   PZKP1                                    (PZKP1+ICR1):=0
003001  566712            ESK     PRRP1                                    IF B=PR-1 THEN SKIP A LINE
                 $                        ELSE GO INITIALIZE NEXT TERM, ICR1=ICR1+1
003002  342777            JP      PINIT2                                   CONTINUE INITIALIZATION
003003  343004            JP      MNINIT                                   END THE INITIALIZATIONS
003004  304000   MNINIT   NOOP                                             END OF THE CMSTR INITIALIZATIONS
                 $
                 $ ***************************************************************
003005  120753            ENTAL   INSFHI
003006  441203            STRAL   RCLIST
003007  440267            STRAL   INGL12                                   ROLL MODEL FOLLOWING
003010  440777            STRAL   PMSENG
003011  400274            STRZ    INGL17
003012  400414            STRZ    IGDS14
003013  401277            STRZ    RCLCLD
003014  401270            STRZ    PTSCLD
003015  401454            STRZ    MABFLG
003016  400666            STRZ    LAMFLG                                   DISABLE CLUTCH SLIP LIGHT ENABLE FLAG
003017  400664            STRZ    A/PFLG                                   HHOLD DETENT ROLL AXIS LIGHT
003020  400665            STRZ    H/RFLG                                   HHOLD DETENT PITCH AXIS LIGHT
003021  400663            STRZ    G/AFLG                                   HHOLD CCLLECTIVE DETENT LIGHT
003022  332307            IJP     CLEAR
003023  121366   CL1      ENTAL   VEZCN
```

```
2896                    *
2897                    ********************************************************************************
2898                    *         THIS IS THE START OF THE SUBROUTINE WHICH DOES THE
2899                    *         MINIMAL COMPENSATOR.  THE INPUTS ARE
2900                    *                 NPLT(PILOT COMMANDS)
2901                    *                 MYS(THE MEASUREMENT VECTOR)
2902                    *         THE OUTPUTS ARE:
2903                    *                 NUT(THE COMMANDS TO THE ACTUATORS)
2904                    *       THE EQUATIONS BEING DONE BY THIS SUBROUTINE ARE
2905                    *
2906                    *           MYDES = MYSCLE * NPLT
2907                    *           NUC = NNK * MYDES
2908                    *           NUI = NUI + NKINT*(MYDES-MYACT)
2909                    *           NZKP1 = NAPIN * NZK  +  NEWIN * MYS
2910                    *           NUR = NCMIN * NZKP1
2911                    *           NUT = NUR + NUC + NUI
2912                    *
2913                    *           WHERE:    NUC== THE SETPOINT CONTROL VECTOR
2914                    *                     NNK-- THE FEEDFORWARD GAIN MATRIX(FOR SETPOINT
2915                    *                           CONTROL
2916                    *                     NPLT-- THE PERTURBATION PILOT INPUT VECTOR
2917                    *                           (PERTURBED ABOUT THE CNTRL AT INITIALIZE)
2918                    *                     NZKP1-- THE PRESENT COMPENSATOR STATE VECTOR
2919                    *                     NANIN== THE COMPENSATOR DYNAMICS MATRIX
2920                    *                     NZK-- THE LAST COMPENSATOR STATE VECTOR
2921                    *                     NEWIN== THE COMPENSATOR INPUT DIST. MATRIX
2922                    *                     MYS-- THE MEAS. VECTOR
2923                    *                     NUR== THE REGULATOR/COMPENSATOR CONTROL VECTOR
2924                    *                     NCMIN-- THE COMPENSATOR OUTPUT MATRIX
2925                    *                     NUT== THE TOTAL PERTURBATION CONTROL
2926                    ********************************************************************************
2927                    *
2928                    *           RESERVE WORDS FOR USE IN THE BRANCHING INSTRUCTIONS
2929                    *
2930  005164  000000  NRWEND   RESERV   RESERV'1D                          ROW COUNTER
2931  005165  000000  NRRM1    RESERV   RESERV'1D                          ORDER OF COMPENSATOR - 1
2932  005166  000000  NRR      RESERV   RESERV'1D                          ORDER OF COMPENSATOR
2933  005167  000000  NRRP1    RESERV   RESERV'1D                          ORDER OF COMPENSATOR + 1
2934  005170  000000  NNCM1    RESERV   RESERV'1D                          NUMBER OF CONTROLS - 1
2935  005171  000000  NNCTRL   RESERV   RESERV'1D                          NUMBER OF CONTROLS
2936  005172  000000  NNMEAS   RESERV   RESERV'1D                          NUMBER OF MEASUREMENTS
2937  005173  000000  CMSTR2   D'0
2938                    *
2939                    *       INITIALIZE BRANCHING VARIABLES SINCE BSR DOESN'T USE
2940                    *       THE SPECIAL REGISTER
2941                    *
2942  005174  507313            ENTSR    13                                USE MEMORY 3 REGISTER
2943  005175  122333            ENTAL    NNC                               AL:=(NNC)
2944  005176  102334            ENTAU    NNP                               AU:=(NNP)
2945  005177  507310            ENTSR    10                                USE MEMORY BANK 0
2946  005200  445171            STRAL    NNCTRL                            NNCTRL:=NNC
2947  005201  717776            ADDADR   -1                                AL:=NNC-1
2948  005202  445170            STRAL    NNCM1                             NNCM1:=NNC-1
2949  005203  445172            STRAU    NNMEAS                            NNMEAS:=NNM
2950  005204  507313            ENTSR    13                                USE MEMORY BAN. 3
2951  005205  122332            ENTAL    NRDATA                            AL:=(NRDATA)
2952  005206  102337            ENTAU    NRR1                              AU:=(NRP1)
```

Longitudinal CAS Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   HOVR43  10/02/84   SECTION   1                                        PAGE   63

005207  507310              ENTSR     1C                                    USE MEMORY BANK 0
005210  445166              STRAL     NRR                                   NRR:=NRCATA
005211  717776              ADDALK    -1                                    AL:=AL-1=NRDATA-1
005212  445165              STRAL     NRRP1                                 NRRP1:=NRCATA-1
005213  465167              STRAU     NRRF1                                 NRRP1:=NRF1
                        *
                        *  ----------SECTION NSTFO--------------
                        *
                        *    THIS SECTION PUTS MEASUREMENTS NEEDED BY THE CONTROLLER
                        *    INTO THE MEASUREMENT VECTOR, NYS.  IT ALSO PUTS THE ACTUAL
                        *    VALUES OF THE DESIRED OUTPUTS INTO THE VECTOR, NYACT.
                        *
                        *    IN BASIC, THE CODE WOULD BE:
                        *
                        *             FOR I=1 TO NM                         ]
                        *                NYS(I)=MEAS(NYSSEL(I))             ]--NSTP01
                        *             NEXT I                                ]
                        *
                        *             FOR I=1 TO NC                         ]
                        *                NYACT(I)=MEAS(NYDSEL(I))           ]--NSTP02
                        *             NEXT I                                ]
                        *
                        *    NYSSEL IS A VECTOR CONTAINING, IN ORDER, THE INDEX NUMBERS
                        *    OF THE SELECTED MEASUREMENTS.  NYDSEL IS A VECTOR CONTAINING,
                        *    IN ORDER, THE INDEX NUMBERS OF THE DESIRED MEASUREMENTS(THOSE
                        *    BEING COMMANDED BY THE PILOT).  THE MEASUREMENTS COME FROM BANK 3.
                        *    THE INDEX NUMBERS AND THE RELATED POSSIBLE MEASUREMENTS ARE
                        *    SHOWN BELOW:
                        *
                        *      INDEX NUMBER              PARAMETER
                        *            1             THETA(+NOSE UP,500/DEG)
                        *            2             PHI(+RWD,500/DEG)
                        *            3             PSI(MAGNETIC,500/DEG)
                        *            4             Q(+NOSE UP,500/(DEG/SEC))
                        *            5             P(+RWD,500/(DEG/SEC))
                        *            6             R(+RIGHT,500/(DEG/SEC))
                        *            7             LONG. ACCEL.(NO GRAV,+FORW,128/(FT/SEC**2))
                        *            8             LAT. ACCEL.(NO GRAV,+RIGHT,128/(FT/SEC**2))
                        *            9             NORMAL ACCEL.(NO GRAV,+UP,128/(FT/SEC**2))
                        *           10             DELTA FORW. VEL.(+FORW,87/(FT/SEC))
                        *           11             VERT. VEL.(+UP,32/(FT/SEC))
                        *           12             BARO. ALT.(16/FT)
                        *
                        *    NOTE THAT THETA AND AIRSPEED ARE PERTURBATION QUANTITIES AND HAVE
                        *    TRIM VALUES SUBTRACTED FROM THEM.*
005214  507313  NSTPO       ENTSR     13                                    USE MEMORY BANK 3
005215  120746              ENTAL     INSTH                                 AL:=ANGTH(PITCH ANGLE)
005216  102344              SUBAL     NTRPTH                                AL:=AL-NTRMTH(PERTURBATION THETA)
005217  442347              STRAL     NCTHET                                NDTHET:=PERTURBATION PITCH ANGLE
                        *                  +NOSE UP, 500/DEG
005220  120753              ENTAL     INSPHI
005221  442350              STRAL     NPHI                                  NPHI IS ROLL,+RWD,500/DEG
005222  120766              ENTAL     INSPSI
005223  442351              STRAL     NPSI                                  NPSI IS HEADING, MAGNETIC,500/DEG
005224  120743              ENTAL     PITR/G
005225  442352              STRAL     NPITRT                                PITCH RATE,+ NOSE UP,500/(DEG/SEC)
005226  120744              ENTAL     RCLR/G
```

Longitudinal CAS Subroutine. (contd)

```
1019A ASSEMBLER, VERSION 1.0   HOVR43  10/02/84   SECTION   1                                     PAGE  64

005227  442353           STRAL    NROLRT         ROLL RATE,+ RWD,500/(DEG/SEC)
005230  120745           ENTAL    YAWR/G
005231  442354           STRAL    NYAWRT         YAW RATE,+RIGHT,500/(DEG/SEC)
005232  120747           ENTAL    ACCLNG
005233  442355           STRAL    NAX            LONG. ACCEL.,NO GRAV,+FORW,128/(FT/SEC**2)
005234  120750           ENTAL    ACCLAT
005235  442356           STRAL    NAY            LAT. ACCEL.,NO GRAV,+RIGHT128/(FT/SEC**2)
005236  120751           ENTAL    ACCNRM
005237  442357           STRAL    NAZ            NORMAL ACCEL.,NO GRAV,+UP,128/(FT/SEC**2)
005240  120752           ENTAL    AIRSPD
005241  162345           SUBAL    NIRPAS         AL:=AIRSPD-TRIM AIRSPEED
005242  442360           STRAL    NCVEL          NCVEL IS PERTURBATION VELOCITY,
                *                 +FORW,8/(FT/SEC)
005243  120765           ENTAL    IVSI           VERT. SPEED,+UP,32/(FT/SEC)
005244  162346           SUBAL    NIRPHD         SUBTRACT HDOT AT ENGAGE
005245  442361           STRAL    NVVEL          PERTURBED VERT. SPEED,+UP,32/(FT/SEC)
005246  120754           ENTAL    BARC
005247  442362           STRAL    NEALT          BARO. ALT., 16/FT
                *
                *  THIS DOES THE NYS SELECTION
                *
005250  507201           ENTICR   1              USE ICR1 AS INDEX I
005251  360001           ENTBK    1              START WITH ICR1=1 SO (I=2)
005252  132323  NSTP01   ENTALB   NYSSEL-1       AL:=(NYSSEL+ICR1-1)
005253  441650           STRAL    TMP            TMP:=NYSSEL(I)
005254  507202           ENTICR   2              USE ICR2 AS INDEX
005255  321650           ENTB     TMP            B:=TMP=NYSSEL(I)
005256  132346           ENTALB   NCTHET-1       AL:=NEAS(NYSSEL(I))=(NDTHET+TMP-1)
005257  507201           ENTICR   1              USE INDEX REGISTER 1 AGAIN
005260  452234           STRALB   NYS-1          AL:=(NYS+ICR1-1)=NYS(I)
005261  565172           PSK      NANEAS         IF B=NANEAS DO NEXT LINE
                *                ELSE ICR1=ICR1+1 AND DO NEXT LINE
005262  345252           JP       NSTP01         CONTINUE
                *
                *  THIS DOES THE NYACT SELECTION
                *
005263  507201           ENTICR   1              USE ICR1 AS INDEX I
005264  360001           ENTBK    1              START WITH ICR1=1 SO (I=1)
005265  132327  NSTP02   ENTALB   NYDSEL-1       AL:=(NYDSEL+ICR1-1)
005266  441650           STRAL    TMP            TMP:=NYDSEL(I)
005267  507202           ENTICR   2              USE ICR2 AS INDEX
005270  321650           ENTB     TMP            B:=TMP=NYDSEL(I)
005271  132346           ENTALB   NCTHET-1       AL:=NEAS(NYDSEL(I))=(NDTHET+TMP-1)
005272  507201           ENTICR   1              USE INDEX REGISTER AGAIN
005273  452252           STRALB   NYACT-1        NYACT(I)=(NYACT+ICR1-1)
005274  565171           PSK      NNCTRL         IF B=NNCTRL SKIP A LINE
                *                ELSE ICR1=ICR1+1 AND DO THE NEXT LINE
005275  345265           JP       NSTP02         GET NEXT NYACT
005276  345277           JP       NSTP03         GO GET PILOT INPUTS
                *
                *  THIS SECION PUTS EXPERIMENTAL PILOT COMMANDS INTO THE NPLT VECTOR
                *
005277  120755  NSTP03   ENTAL    EPLCN          AL:=EP LONG STICK
005300  504600           LRTAL    0              SCALE THE INPUT OF LONG CYLIC
005301  442231           STRAL    NPLT           NPLT(1)=LONG STICK,1024PER IN,+AFT
005302  120760           ENTAL    EPHPAV         AL:=EP COLLECTIVE
005303  504600           LRTAL    0              SCALE THE INPUT OF COLLECTIVE
```

Longitudinal CAS Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   HOVR43  10/02/84   SECTION   1                                             PAGE  65

005304  442232                STRAL    NFLT+1                  NPLT(2)=COLLECTIVE,1024 PER IN,+UP
005305  120756                ENTAL    EFLAT                   AL:=EP LATERAL STICK
005306  504600                LRTAL    0                       SCALE THE LATERAL STICK
005307  442233                STRAL    NFLT+2                  NPLT(3)=LATERAL CYCLIC,1024 PER IN,+RIGHT
005310  120757                ENTAL    EFYAW                   AL:=EP YAW COMMANDS
005311  504600                LRTAL    0                       SCALE THE PEDALS
005312  442234                STRAL    NFLT+3                  NPLT(4)=YAW CONTROL,1024 PER IN,+RIGHT??
                      *
                      *  ----------SECTION NSTP1--------------
                      *
                      *    THIS SECTION PUTS NPLT * COMMAND SCALING INTO THE NYDES
                      *    (DESIRED OUTPUT VECTOR)
                      *
                      *              NYDES = NYSCLE * NPLT
                      *
                      *          WHERE NYSCLE IS THE DIAGONAL SCALING MATRIX
005313  507310  NSTP1   ENTSR    10                      USE MEMORY BANK 0
005314  507201          ENTICR   1                       USE ICR1 AS THE INDEX REG.(B)
005315  360000          ENTBK    0                       INITIALIZE THE B REG TO 0
005316  507313          ENTSR    13                      USE MEMORY BANK 3
005317  132231  NSTP10  ENTALB   NPLT                    AL:=(NPLT+ICR1)
005320  252257          PULALB   NYSCLE                  A:=AL*(NYSCLE + ICR1)
005321  504702          LRTA     2                       NSTP1 SCALING WAS 2**16
005322  472247          STRAUB   NYDES                   STORE RESULT IN (NYDES+ICR1),SCALING IS 2**16
005323  565170          BSK      NNCM1                   IF ICR1=NNCM1(NC-1) THEN SKIP NEXT LINE
                      *                ELSE DO NEXT LINE AND ICR1=ICR1+1
005324  345317          JP       NSTP10                  GO SCALE THE NEXT CONTROL
005325  345326          JP       NSTP2                   GO TO THE INTEGRAL CONTROL SECTION
                      *
                      *  ***********SECTION NSTP2************
                      *
                      *    THIS SECTION DOES THE INTEGRAL CONTROL CALCULATION FOR
                      *    THE CONTROLLER SUBROUTINE.  FOR THE INTEGRATORS WE USE
                      *    FOLLOWING EQUIVALENT BASIC COMMANDS:
                      *
                      *           NUI = NUI + NFINT * (NYDES-NYACT)
                      *
                      *        WHERE NUI IS THE OUTPUT OF THE INTEGRATOR.
005326  507310  NSTP2   ENTSR    10                      USE MEMORY BANK 0
005327  507201          ENTICR   1                       USE ICR1 AS THE INDEX REG.(B)
005330  360000          ENTBK    0                       INITIALIZE THE B REG. TO 0
005331  400002          STRZ     2                       ICR2=0, THE COLUMN INDEX
005332  400003          STRZ     3                       ICR3=0, THE ROW INDEX
005333  507313          ENTSR    13                      USE MEMORY BANK 3
005334  122333          ENTAL    NNC                     AL:=NUMBER OF CONTROLS
005335  717776          ADDALK   -1                      AL:=NNC-1
005336  507310          ENTSR    10                      USE MEMORY BANK 0
005337  445104          STRAL    NRWEND                  NRWEND:=NNC-1
005340  507313          ENTSR    13                      USE MEMORY BANK 3
005341  700000  NNXRW4  ENTALK   0                       AL:=0
005342  507400          XFRLU                            AU:=AL=0
005343  500800  NNXCL4  STRA     TPP                     TMP1=A
005344  031650
005345  507202          ENTICR   2                       GET THE COLUMN INDEX
005346  370001          ENTBKB   1                       INCREMENT COLUMN INDEX(ICR2=ICR2+1)
005347  132246          ENTALB   NYDES-1                 AL:=(NYDES+ICR2-1)
005350  172252          SUBALB   NYACT-1                 AL:=(NYDES+ICR2-1)-(NYACT+ICR2-1)
```

Longitudinal CAS Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION   1                                        PAGE   56

005351  507201              ENTICR   1          USE ICR1 AS INDEX REG.(ELEMENT INDEX)
005352  252201              MULALB   NKINT      A:=AL * (NKINT+ICR1)
005353  201650              ADDA     TPP        A:=A + (TPP,TPP+1)
005354  505300              SKPNOV              CHECK FOR OVERFLOW
005355  504000              NOOP                CALL OVERFLOW HANDLER
005356  565164              ESK      NRWEND     IF ICR1=NRWEND THEN SKIP NEXT LINE
                *                     ELSE DO NEXT LINE AND ICR1=ICR1+1
005357  345343              JP       NNXCL4     JUMPS TO NEXT COLUMN OF NKINT
                *           AND NEXT ROW ELEMENT OF (NYDES-NYACT)
005360  370001              ENTBRB   1          ICR1=ICR1+1
005361  507203              ENTICR   3          USE ICR3 AS INDEX REG.(ROW INDEX)
005362  504703              ENTA     SC         SCALING FOR NSTP2,2**15
005363  507500              XFRUL               PUT AU(THE RESULT) INTO AL
005364  152211              ADDALB   NUI        AL:=AL+(NUI+ICR1)
005365  452211              STRALB   NUI        NUI+ICR3:=(AL)=(NUI+ICR3)+AL
005366  565170              ESK      NNCP1      IF ICR3=NNCP1 THEN SKIP NEXT LINE
                *                     ELSE DO NEXT LINE AND ICR3=ICR3+1
005367  345371              JP       NRSET4     GO RESET THE COUNTER
005370  345401              JP       NSTF3      GO TO NEXT MATRIX MULT IF DONE
005371  507202  NRSET4      ENTICR   2          SAVE ROW INDEX AND START USING COLUMN INDEX
005372  360000              ENTBK    0          RESET COLUMN INDEX TO ZERO
005373  507310              ENTSR    IC         BANK 0 TO USE NRWEND AND NNCTRL
005374  125164              ENTAL    NRWEND     AL:=NRWEND
005375  145171              ADDAL    NNCTRL     AL:=NRWEND+NNCTRL=NRWEND+# OF COLUMNS
005376  445164              STRAL    NRWEND     NRWEND:=NRWEND+NNCTRL
005377  507313              ENTSR    13         BANK 3 FOR DATA
005400  345341              JP       NNXRW4     DO THE NEXT ROW
                *
                *   ----------SECTION NSTF3--------
                *
                *   FIRST DO THE SET POINT GENERATOR MATRIX MULTIPLICATION
                *                NUC = NNK * NYDES
                *                WHERE NUC = NNC X NNC   AND  NYDES = NNC X 1
005401  507310  NSTF3       ENTSR    IC         USE MEMORY BANK 0
005402  507201              ENTICR   1          USE ICR1 AS THE INDEX REG.(B)
005403  360000              ENTBK    0          INITIALIZE THE B REG. TO 0
005404  400002              STRZ     2          ICR2=0, THE COLUMN INDEX
005405  400003              STRZ     3          ICR3=0, THE ROW INDEX
005406  507313              ENTSR    13         USE MEMORY BANK 3
005407  122333              ENTAL    NNC        AL:=NUMBER OF CONTROLS
005410  717776              ADDALK   -1         AL:=NNC-1
005411  507310              ENTSR    SC         USE MEMORY BANK 0
005412  445164              STRAL    NRWEND     NRWEND:=NNC-1
005413  507313              ENTSR    13         USE MEMORY BANK 3
005414  700000  NNXRW1      ENTALK   0          AL:=0
005415  507400              XFRCU               AU:=AL=0
005416  500600  NNXCL1      STRA     TPP        TPP:=A
005417  031050
005420  507202              ENTICR   2          GET THE COLUMN INDEX
005421  370001              ENTBRB   1          INCREMENT COLUMN INDEX(ICR2=ICR2+1)
005422  132246              ENTALB   NYDES-1    AL:=(NYDES+ICR2-1)
005423  507201              ENTICR   1          USE ICR1 AS INDEX REG.(ELEMENT INDEX)
005424  252265              MULALB   NNK        A:=AL * (NNK+ICR1)
005425  201650              ADDA     TPP        A:=A + (TPP,TPP+1)
005426  505300              SKPNOV              CHECK FOR OVERFLOW
005427  504000              NOOP                CALL OVERFLOW HANDLER
005430  565164              ESK      NRWEND     IF ICR1=NRWEND THEN SKIP NEXT LINE
```

Longitudinal CAS Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   HCVR43  10/02/84   SECTION   1                                                  PAGE  67

                       *                              ELSE DO NEXT LINE AND ICR1=ICR1+1
005431  345416                JP        NNXCL1                                    JUMPS TC NEXT COLUMN OF NNK
                       *                              AND NEXT RCW ELEMENT OF NFLT
005432  370001                ENTBKB    1                                         ICR1=ICR1+1
005433  507203                ENTICR    3                                         USE ICR3 AS INDEX REG.(ROW INDEX)
005434  504703                LRTA      3                                         SCALING FCR NSTP3,2**15
005435  472215                STRAUB    NUC                                       NUC+ICR3:=(AU)
005436  565170                BSK       NNCPI                                     IF ICR3=NNCPI THEN SKIP NEXT LINE
                       *                              ELSE DO NEXT LINE AND ICR3=ICR3+1
005437  345441                JP        NRSET1                                    GO RESET THE COUNTER
005440  345451                JP        NSTP4                                     GO TO NEXT MATRIX MULT IF DONE
005441  507202    NRSET1      ENTICR    2                                         SAVE ROW INDEX AND START USING COLUMN INDEX
005442  360000                ENTBK     0                                         RESET CCLUMN INDEX TO ZERO
005443  507310                ENTSR     S0                                        BANK 0 TO USE NRWEND AND NNCTRL
005444  125164                ENTAL     NRWEND                                    AL:=NRWENC
005445  145171                ADDAL     NNCTRL                                    AL:=NRWENC+NNCTRL=NRWEND+# OF COLUMNS
005446  445164                STRAL     NRWEND                                    NRWENC:=NRWENC+NNCTRL
005447  507313                ENTSR     I3                                        BANK 3 FOR DATA
005450  345414                JP        NNXRW1                                    CO THE NEXT ROW
                       *
                       *   --------------------------SECTION NSTP4--------
                       *
                       *              THIS IS THE 2ND MATRIX MULTIPLY IN THE CMSTR2
                       *              SUBROUTINE, I.E.
                       *                         NEMIN * NYS
005451  507310    NSTP4       ENTSR     S0                                        USE MEMCRY BANK 0
005452  507201                ENTICR    1                                         USE ICR1 AS THE INDEX REG.(B)
005453  360000                ENTBK     0                                         INITIALIZE THE B REG. TO 0
005454  400002                STRZ      2                                         ICR2=0, THE COLUMN INDEX
005455  400003                STRZ      3                                         ICR3=0, THE ROW INDEX
005456  507313                ENTSR     I3                                        USE MEMCRY BANK 3
005457  122334                ENTAL     NNM                                       AL:=NUMBER OF MEASUREMENTS
005460  717776                ADDALK    -1                                        AL:=NNM-1
005461  507310                ENTSR     S0                                        USE MEMCRY BANK 0 SINCE NRWEND IS THERE
005462  445164                STRAL     NRWEND                                    NRWEND:=NNM-1
005463  507313                ENTSR     I3                                        BACK TO MEMORY BANK 3 FOR DATA
005464  700000    NNXRW2      ENTALK    0                                         AL:=0
005465  507400                XFRLU                                               AU:=AL=0
005466  500600    NNXCL2      STRA      TPP                                       TPP:=A
005467  031650
005470  507202                ENTICR    2                                         GET THE CCLUMN INDEX
005471  370001                ENTBKB    1                                         INCREMENT COLUMN INDEX(ICR2=ICR2+1)
005472  132234                ENTALB    NYS-1                                     AL:=(NYS+ICR2-1)
005473  507201                ENTICR    1                                         USE ICR1 AS INDEX REG.(ELEMENT INDEX)
005474  252302                MULALB    NEMIN                                     A:=AL * (NBMIN+ICR1)
005475  201650                ADDA      TPP                                       A:=A + (TPP,TMP+1)
005476  505300                SKPNOV                                              CHECK FCR OVERFLOW
005477  506000                NOOP                                                CALL OVERFLOW HANDLER
005500  565164                BSK       NRWEND                                    IF ICR1=NRWEND THEN SKIP NEXT LINE
                       *                              ELSE DO NEXT LINE AND ICR1=ICR1+1
005501  345466                JP        NNXCL2                                    JUMPS TC NEXT COLUMN OF NBMIN
005502  370001                ENTBKB    1                                         ICR1=ICR1+1
005503  507203                ENTICR    3                                         USE ICR3 AS INDEX REG.(ROW INDEX)
005504  504705                LRTA      5C                                        NSTP4 SCALING IS 2**13
005505  472274                STRAUB    NZKP1                                     NZKP1+ICR3:=(AU)
005506  565165                BSK       NNMP1                                     IF ICR3=NNM THEN SKIP NEXT LINE
                       *                          ELSE DO NEXT LINE AND ICR3=ICR3+1
```

Longitudinal CAS Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION   1                                                PAGE  68

005507  345511            JP       NRSET2                              GO RESET THE COUNTER
005510  345521            JP       NSTP5                               GO TO NEXT MATRIX MULT IF DONE
005511  507202  NRSET2    ENTICR   2                                   SAVE ROW INDEX AND START USING COLUMN INDEX
005512  360000            ENTBK    0                                   RESET COLUMN INDEX TO ZERO
005513  507310            ENTSR    10                                  BANK 0 TO USE NPWEND AND NNMEAS
005514  125164            ENTAL    NPWEND                              AL:=NPWEND
005515  145172            ADDAL    NNMEAS                              AL:=NPWEND+NNMEAS
005516  445164            STRAL    NPWEND                              NPWEND:=NPWEND+NNMEAS=NPWEND+NOF COLUMNS
005517  507313            ENTSR    13                                  BACK TO MEMORY BANK 3 FOR DATA
005520  345404            JP       NNXRW2                              DO THE NEXT ROW
                *
                *  ----------------SECTION NSTP5--------------.
                *
                *         THIS STEP(NSTP5) DOES THE MATRIX MULTIPLICATION
                *         DESCRIBED BY
                *                NAMIN*NZK
                *         WHERE NAMIN IS THE MINIMAL FORM OF THE COMPENSATOR
                *          DYNAMICS MATRIX
005521  507202  NSTP5     ENTICR   2                                   SELECT ICR2(J=ICR2)
005522  360000            ENTBK    0                                   ICR2:=0
005523  507201            ENTICR   1                                   SELECT ICR1(I=ICR1)
005524  360000            ENTBK    0                                   ICR1:=1
005525  565166  NSTP50    BSK      NRR                                 IF ICR=NR SKIP ONE
                *                      LINE ELSE ICR1=ICR1+1(I=I+1) AND  DO NEXT LINE
005526  345530            JP       NSTP51                              GO CHECK FOR TYPE OF MINIMAL BLOCK
005527  345566            JP       NSTP54                              GO TO SUMMATION WITH NZKP1
005530  507202  NSTP51    ENTICR   2                                   USE ICR2 AS INDEX,J
005531  132335            ENTALB   NADESC                              AL:=(NADESC+ICR2)
005532  022375            CMAL     NONE                                COMPARE AL,I.E. NADESC(J), WITH 1
005533  615537            JPEQ     NSTP52                              GOTO NSTP52 IF NADESC(J)=1
005534  022376            CMAL     NTWO                                COMPARE AL TO 2
005535  615546            JPEQ     NSTP53                              GOTO NSTP53 IF NADESC(J)=2
005536  504000            NOOP                                         A CALL TO AN ERROR HANDLER GOES HERE
                *    THIS IS THE SECTION FOR 1X1 BLOCKS OF AMIN
005537  370001  NSTP52    ENTBKB   1                                   ICR2=ICR2+1,I.E. J=J+1
005540  507201            ENTICR   1                                   ICR1 IS THE INDEX REG.
005541  132276            ENTALB   NAMIN-1                             AL:=AL*(NAMIN+ICR1-1)
005542  252270            PULALB   NZK-1                               A:=AL*(NZK+ICR1-1)
005543  504703            LRTA     3                                   SCALING FOR NSTP52 IS 2**15
005544  472362            STRAUB   TMPNZ-1                             TMPNZ+1:=AU=A(I)*NZK(I),SCALING IS 2**15
005545  345525            JP       NSTP50                              JUMP BACK TO GET NEXT AMIN DESCRIPTOR
                *    THIS IS THE SECTION FOR 2X2 BLOCKS OF AMIN
005546  370001  NSTP53    ENTBKB   1                                   ICR2=ICR2+1,I.E. J=J+1
005547  507201            ENTICR   1                                   ICR1 IS NOW THE INDEX REG.
005550  132271            ENTALB   NZK                                 AL:=(NZK+ICR1)
005551  452362            STRALB   TMPNZ-1                             TMPNZ+ICR1-1:=(NZK+ICR1),I.E.
                *                                 TMPNZ(I)=NZK(I+1)
005552  132276            ENTALB   NAMIN-1                             AL:=(NAMIN+ICR1-1)
005553  252270            PULALB   NZK-1                               A:=AL*(NZK+ICR1-1)
005554  504703            LRTA     3                                   SCALING FOR NSTP53 IS 2**15
005555  461650            STRAU    TMP                                 TMP:=AU, I.E. TMP=NAMIN(I)*NZK(I),SCALING IS
                                                                       2**15
005556  132277            ENTALB   NAMIN                               AL:=(NAMIN+ICR1)
005557  252271            PULALB   NZK                                 A:=AL*(NZK+ICR1);I.E.
005560  504703            LRTA     3                                   SCALING FOR NSTP53 IS 2**15
                *                                 A=NAMIN(I+1)*NZK(I+1)
005561  507500            XFRUL                                        PUT AU INTO AL
```

Longitudinal CAS Subroutine. (contd)

```
IBIYA ASSEMBLER, VERSION 1.0   NOVR43  10/02/84   SECTION  1                                                     PAGE  69

 005562  141650            ADDAL    TPP                                  AL:=AL+INF,I.E. AL=A(I)*Z(I)+A(I+1)*Z(I+1)
 005563  452363            STHALB   TPPNZ                                TPPNZ+ICRI:=AL,SCALING IS 2**15
 005564  370001            ENTBKB   1                                    ICR1=ICR1+1 BUT THIS MEANS I=I+2
                *                                 SINCE NSTF50 INCREMENTED ICR1 ONCE ALREADY
 005565  345525            JP       NSTF50                               GO GET THE NEXT AMIN DESCRIPTOR
 005566  360001  NSTP54    ENTBK    1                                    ICR1:=1
                *           THIS SECTION ADDS NBMIN*NYS + AAPIN*NZK
                *           AND PUTS THE RESULT IN THE NZKF1 VECTOR
 005567  132273  NSTP55    ENTALB   NZKF1-1                              AL:=(NZKP1+ICR1-1)
 005570  152362            ADDALB   TPPNZ-1                              AL:=AL+(TPPNZ+ICR1-1)
 005571  452273            STRALB   NZKP1-1                              NZKP1+ICR1-1:=AL
 005572  565166            ESK      NRR                                  IF ICR1=NR+1 THEN SKIP ONE LINE
                *                                 ELSE ICR1=ICR1+1 AND DO NEXT LINE
 005573  345567            JP       NSTF55                               CONTINUE
 005574  345575            JP       NSTF6                                GOTO NEXT STEP
                *
                *  -------------SECTION NSTP( ---------
                *
                *             THIS IS THE 4TH MATRIX MULTIPLY IN THE CMSTR2
                *             SUBROUTINE, I.E.
                *                           NUR = ICMIN * NZKF1
 005575  507310  NSTP6     ENTSR    IC                                   USE MEMORY BANK 0
 005576  507201            ENTICR   1                                    USE ICR1 AS THE INDEX REG.(B)
 005577  360000            ENTBK    0                                    INITIALIZE THE E REG. TO 0
 005600  400002            STRZ     2                                    ICR2=0, THE COLUMN INDEX
 005601  400003            STRZ     3                                    ICR3=0, THE ROW INDEX
 005602  507313            ENTSR    13                                   USE MEMORY BANK 3
 005603  122332            ENTAL    NRDATA                               AL:=NUM OF COMPENSATOR STATES
 005604  717776            ADDALK   -1                                   AL:=NRDATA-1
 005605  507310            ENTSR    10                                   USE MEMORY BANK 0 SINCE NRWEND IS THERE
 005606  445164            STRAL    NRWEND                               NRWEND:=NRDATA-1
 005607  507313            ENTSR    13                                   BACK TO MEMORY BANK 3 FOR DATA
 005610  700000  NNXRN3    ENTALK   0                                    AL:=0
 005611  507400            XFRLU                                         AU:=AL=0
 005612  500600  NNXCL3    STRA     TPP                                  TMP:=A
 005613  031650
 005614  507202            ENTICR   2                                    GET THE COLUMN INDEX
 005615  370001            ENTBKH   1                                    INCREMENT COLUMN INDEX(ICR2=ICR2+1)
 005616  132273            ENTALB   NZKF1-1                              AL:=(NZKP1+ICR2-1)
 005617  507201            ENTICR   1                                    USE ICR1 AS INDEX REG.(ELEMENT INDEX)
 005620  252316            PULALB   NTCPIN                               A:=AL * (NTCMIN+ICR1)
 005621  201650            ADDA     TPP                                  A:=A + (TPP,TMP+1)
 005622  505300            SKPNOV                                        CHECK FOR OVERFLOW
 005623  504000            NOCP                                          CALL OVERFLOW HANDLER
 005624  565164            ESK      NRWEND                               IF ICR1=NRWEND THEN SKIP NEXT LINE
                *                                 ELSE DO NEXT LINE AND ICR1=ICR1+1
 005625  345612            JP       NNXCL3                               JUMPS TO NEXT COLUMN OF NTCMIN
                *                                 AND NEXT ROW ELEMENT OF NZKP1
 005626  370001            ENTBKB   1                                    ICR1=ICR1+1
 005627  507203            ENTICR   3                                    USE ICR3 AS INDEX REG.(ROW INDEX)
 005630  504700            LRTA     0                                    SCALING FOR NSTP6 IS 2**18
 005631  472221            STRAUB   NUR                                  NUR+ICR3:=(AU),SCALING IS 2**18
 005632  565170            ESK      NNCP1                                IF ICR3=NNCN1 THEN SKIP NEXT LINE
                *                                 ELSE DO NEXT LINE AND ICR3=ICR3+1
 005633  345635            JP       NRSET3                               GO RESET THE COUNTER
 005634  345643            JP       NSTF7                                GO TO NEXT MATRIX MULT IF DONE
 005635  507202  NRSET3    ENTICR   2                                    SAVE ROW INDEX AND START USING COLUMN INDEX
```

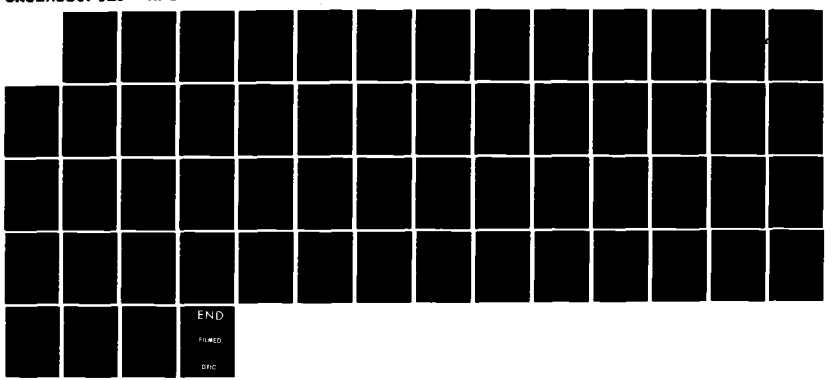
Longitudinal CAS Subroutine. (contd)

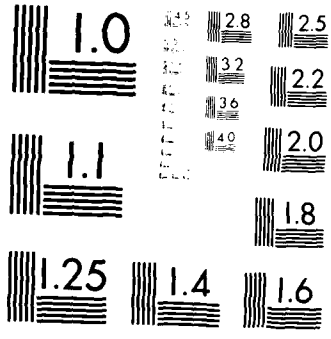
```
1819A ASSEMBLER, VERSION 1.C   MOVR43  10/02/14   SECTION   1                                                   PAGE  70

005636  360000            ENTBK    0                                RESET CCLUMN INDEX TO ZERO
005637  507310            ENTSR    IC                               USF BANK 0 SINCE NRWEND AND NNCTRL ARE THERE
005640  125164            ENTAL    NRWEND                           AL:=NRWENC
005641  145166            ADDAL    NRR                              AL:=NRWENC+NRR=NRWEND+NCF COLUMNS
005642  445164            STRAL    NRWEND                           NRWENC:=NRWENC+NNCTRL
005643  507313            ENTSR    13                               USE BANK 3 FOR CATA
005644  345610            JP       NAXRW3                           CO THE NEXT RCW
                 *
                 *   ------------SECTION NSTP7---------
                 *
                 *             THIS STEP DCES THE ADDITION NUT = NUC + NUR + NUI
                 *
005645  507201  NSTP7     ENTICR   1                                USE ICR1
005646  360001            ENTBK    1                                (ICR1I=0) MEANS (I=1)
005647  132214  NSTP70    ENTALB   NUC-1                            AL:=(NUC+ICR1-1)
005650  152220            ADDALB   NUR-1                            AL:=AL+(NUR+ICR1-1)
005651  152210            ADDALB   NUI-1                            AL:=AL+(NUI+ICR1-1)
005652  505300            SKPNOV                                    SKIP A LINE IF NO OVERFLOW
005653  504000            NOOP                                      CALL CVERFLOW HANDLER
005654  452224            STRALB   NUT-1                            NUTI=AL, NUT(I)=NUC(I)+NUR(I)
005655  565171            ESK      NNCTRL                           IF BSK=NNCTRL SKIP THE NEXT LINE
                 *                                 ELSE ICR1I=ICR1+1 AND CO NEXT LINE
005656  345647            JP       NSTP70                           CONTINUE
005657  345660            JP       NSTP8                            WRITE THE DATA CUT
                 *
                 *   ----------------SECTICN NSTP8-----------
                 *
                 *             HERE GOES THE SECTION(NSTP8) WHICH WRITES NUT(ACTUATCR
                 *             CCMMANDS) TC THE LOCATION  WHICH SENDS NUT TC THE
                 *             ACTUATORS.
                 *                                NSTP8
005660  122225  NSTP8     ENTAL    NUT                              AL:=NUT(1),LONG CYCLIC
005661  242340            MULAL    NLNGSC                           SCALE FCR OUTPUT(1IN/140.85BITS)
005662  460072            STRAU    OBP1                             STORE NUT(1) TO ECS PITCH COMMAND
005663  122226            ENTAL    NUT+1                            AL:=NUT(2),COLLECTIVE
005664  242341            MULAL    NCOLSC                           SCALE THE OUTPUT(1IN/206.24BITS)
005665  460075            STRAU    OBF4                             STORE NUT(2) TO ECS COLLECTIVE COMMAND
                 *            ENTAL'NUT+2 AL:=NUT(3), LATERAL CYCLIC
                 *            MULAL'NLATEC     SCALE THE OUTPUT(1IN/218.33BITS)
                 *            STRAU'OBF2  STORE NUT(3) TC ECS ROLL CCMMAND
                 *            ENTAL'NUT+3 AL:=NUT(4), DIRECTIONAL CCMMAND
                 *           MULAL'NTAWSC      SCALE THE CUTPUT(1IN/280.93BITS)
                 *            STRAU'OBF3  STORE NUT(4) TC ECS YAW CCMMANC
                 *
                 *   ------------SECTION NSTP9
                 *
                 *             THIS SECTION CF CMSTR2 SETS THE PRESENT COMPENSATOR
                 *             STATES(NZKP1) TO OLD STATES(NZK)
                 *
005666  507201  NSTP9     ENTICR   1                                USE ICR1 AS INDEX
005667  360001            ENTBK    1                                ICR1=0  (I=0)
005670  132273  NSTP90    ENTALB   NZKP1-1                          AL:=(NZKP1+ICR1)
005671  452270            STRALB   NZK-1                            NZK+ICR1-1:=AL
005672  565166            BSK      NRR                              IF ICR1=NRR THEN SKIP 1 LINE
                 *                                ELSE ICR1=ICR1+1  AND DO NEXT LINE
005673  345670            JP       NSTP90                           CONTINUE
005674  555173            IJP      CPSTR2                           END THE CCNTROLLER SUBROUTINE NUMBER 2
```

Hover Controller Subroutine.

```
1819A ASSEMBLER, VERSION 1.0   HOVR43  10/02/84   SECTION   1                                                  PAGE   86

                      *
                      * ************************************************************************
                      *          THIS IS THE START OF THE SUBROUTINE WHICH DOES THE
                      *          PINTNAL COMPENSATOR FOR THE HOVER CONTROLLER.
                      *          THE CONTROL STRATEGY IS "VELOCITY COMMAND/POSITION HOLD"
                      *          THE INPUTS TO THE SUBROUTINE ARE
                      *                 PPLT(PILOT COMMANDS)
                      *                 PYS(THE MEASUREMENT VECTOR)
                      *          THE OUTPUTS ARE:
                      *                 PUT(THE COMMANDS TO THE ACTUATORS)
                      *         THE EQUATIONS BEING DONE BY THIS SUBROUTINE ARE
                      *
                      *            P?DPID = P?DP + P?DI + P?DD + P?DOLD
                      *                  WHERE ? IS X, Y, AND Z.
                      *                  NOTE: THIS EQUATION IS DONE ONLY DURING
                      *                        POSITION HOLD
                      *            PYDES = PYSCLE * (PPLT+P?DPID)
                      *            PUC = PNK * PYDES
                      *            PZKP1 = PPIN * PZK   +  PBMIN * PYS
                      *            PUR = PCMIN * PZKP1
                      *            PUT = PUR + PUC
                      *
                      *             WHERE:    P?DP-- PROPORTIONAL CONTROL
                      *                       P?DI-- INTEGRAL CONTROL
                      *                       P?DD-- DERIVATIVE CONTROL
                      *                       P?DPID-- TOTAL PID CONTROL
                      *                       PUC-- THE SETPOINT CONTROL VECTOR
                      *                       PNK-- THE FEEDFORWARD GAIN MATRIX(FOR SETPOINT
                      *                             CONTROL
                      *                       PPLT-- THE PERTURBATION PILOT INPUT VECTOR
                      *                              (PERTURBED ABOUT THE CNTRL AT INITIALIZE)
                      *                       PZKP1-- THE PRESENT COMPENSATOR STATE VECTOR
                      *                       PAMIN-- THE COMPENSATOR DYNAMICS MATRIX
                      *                       PZK-- THE LAST COMPENSATOR STATE VECTOR
                      *                       PBMIN-- THE COMPENSATOR INPUT DIST. MATRIX
                      *                       PYS-- THE MEAS. VECTOR
                      *                       PUR-- THE REGULATOR/COMPENSATOR CONTROL VECTOR
                      *                       PCMIN-- THE COMPENSATOR OUTPUT MATRIX
                      *                       PUT-- THE TOTAL PERTURBATION CONTROL
                      * ************************************************************************
                      *
                      *            RESERVE WORDS FOR USE IN THE BRANCHING INSTRUCTIONS
                      *
     006711  000000        PNMEND     RESERV     RESERV'10                                  NUM COUNTER
     006712  000000        PRRM1      RESERV     RESERV'10                                  ORDER OF COMPENSATOR - 1
     006713  000000        PRR        RESERV     RESERV'10                                  ORDER OF COMPENSATOR
     006714  000000        PRRP1      RESERV     RESERV'10                                  ORDER OF COMPENSATOR + 1
     006715  000000        PNCM1      RESERV     RESERV'10                                  NUMBER OF CONTROLS - 1
     006716  000000        PNCTRL     RESERV     RESERV'10                                  NUMBER OF CONTROLS
     006717  000000        PNMEAS     RESERV     RESERV'10                                  NUMBER OF MEASUREMENTS
     006720  000000        CMSTR4     0'0
                      *
                      *         INITIALIZE BRANCHING VARIABLES SINCE BSK DOESN'T USE
                      *         THE SPECIAL REGISTER
                      *
     006721  507313               ENTSM      S3                                             USE MEMORY 3 REGISTER
     006722  123136               ENTAL      PNC                                            AL:=(PNC)
```

AD-A151 946 A MODERN CONTROL DESIGN METHODOLOGY WITH APPLICATION TO 3/3
THE CH-47 HELICOPTER(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH R D HOLDRIDGE JAN 85
UNCLASSIFIED AFIT/CI/NR-85-33D F/G 1/4 NL





```
 10:19  ASSEMBLER, VERSION 1.0   NOVR43  10/02/84   SECTION   1                                                    PAGE   87

 4237  006723  103137          ENTAU   PNP                                AU1=(PNP)
 4238  006724  507310          ENTSR   IC                                 USE MEMORY BANK 0
 4239  006725  446716          STRAL   PNCTRL                             PNCTRL1=PNC
 4240  006726  717770          ADDALK  -1                                 AL1=PNC-1
 4241  006727  446715          STRAL   PNCM1                              PNCM11=PNC-1
 4242  006730  446717          STRAU   PNMEAS                             PNMEAS1=PNM
 4243  006731  507313          ENTSR   13                                 USE MEMORY BANK 3
 4244  006732  123135          ENTAL   PR                                 AL1=(PR)
 4245  006733  103144          ENTAU   PRR1                               AU1=(PRR1)
 4246  006734  507310          ENTSR   IC                                 USE MEMORY BANK 0
 4247  006735  446713          STRAL   PRR                                PRR1=PR
 4248  006736  717770          ADDALK  -1                                 AL1=AL*1+PR-1
 4249  006737  446712          STRAL   PRRM1                              PRRM11=PR-1
 4250  006740  446714          STRAU   PRMM1                              PRMM11=PRM1
 4251                  *
 4252                  *  ----------SECTION PSIPC---------------
 4253                  *
 4254                  *    THIS SECTION PUTS MEASUREMENTS NEEDED BY THE CONTROLLER
 4255                  *    INTO THE MEASUREMENT VECTOR, PYS.
 4256                  *
 4257                  *    IN BASIC, THE CODE WOULD BE:
 4258                  *
 4259                  *              FOR I=1 TO NM                      )
 4260                  *                PYS(I)=MEAS(PYSSEL(I))           )<<PSIP01
 4261                  *              NEXT I                             )
 4262                  *
 4263                  *    PYSSEL IS A VECTOR CONTAINING, IN ORDER, THE INDEX NUMBERS
 4264                  *    OF THE SELECTED MEASUREMENTS.  THE MEASUREMENTS COME FROM BANK 3.
 4265                  *    THE INDEX NUMBERS AND THE RELATED POSSIBLE MEASUREMENTS ARE
 4266                  *    SHOWN BELOW:
 4267                  *
 4268                  *       INDEX NUMBER                  PARAMETER
 4269                  *            1              THETA(+NOSE UP,500/DEG)
 4270                  *            2              PHI(+RWD,500/DEG)
 4271                  *            3              PSI(MAGNETIC,500/DEG)
 4272                  *            4              Q(+NOSE UP,500/(DEG/SEC))
 4273                  *            5              P(+RWD,500/(DEG/SEC))
 4274                  *            6              R(+RIGHT,500/(DEG/SEC))
 4275                  *            7              LONG. ACCEL.(NO GRAV,+FORW,128/(FT/SEC**2))
 4276                  *            8              LAT. ACCEL.(NO GRAV,+RIGHT,128/(FT/SEC**2))
 4277                  *            9              NORMAL ACCEL.(NO GRAV,+UP,128/(FT/SEC**2))
 4278                  *           10              DELTA FORW. VEL.(+FORW,8/(FT/SEC))
 4279                  *           11              DELTA VERT. VEL.(+UP,32/(FT/SEC))
 4280                  *           12              BARO. ALT.(16/FT)
 4281                  *           13              X POSITION(+ FORW., 16/FT) IN HEADING INERTIAL AXIS
 4282                  *           14              Y POSITION(+ RIGHT, 16/FT) IN HEADING INERTIAL AXIS
 4283                  *           15              Z POSITION(+ DOWN, 16/FT) IN HEADING INERTIAL AXIS
 4284                  *           16              X INERTIAL VEL.(+ FORW., 32/FPS) IN HEADING INERTIAL AXI
 4285                  *
 4286                  *           17              Y INERTIAL VEL.(+ RIGHT, 32/FPS) IN HEADING INERTIAL AXI
 4287                  *
 4288                  *           18              Z INERTIAL VEL.(+ DOWN, 32/FPS) IN HEADING INERTIAL AXIS
 4289                  *           19              X POSITION(+ FORW., 16/FT) IN RUNWAY FRAME
 4290                  *           20              Y POSITION(+ RIGHT, 16/FT) IN RUNWAY FRAME
 4291                  *           21              X INERTIAL VEL.(+ FORW., 32/FPS) IN RUNWAY FRAME
 4292                  *           22              Y INERTIAL VEL.(+ RIGHT, 32/FPS) IN RUNWAY FRAME
 4293                  *
 4294                  *    NOTE THAT THETA, VEPT, VEL., AND AIRSPEED ARE PERTURBATION QUANTITIES AND MAY
```

Hover Controller Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   HOVR83  10/02/84   SECTION   1                                        PAGE  88

                        $
006741  307313   PSIPO    ENTSR     T3                              USE MEMORY BANK 3
006742  120746            ENTAL     INSTH                           AL:=ANGTH(PITCH ANGLE)
006743  163151            SUBAL     PTRPTH                          AL:=AL-PTRPTH(PERTURBATION THETA)
006744  443202            STRAL     PCTHET                          PDTHET:=PERTURBATION PITCH ANGLE
                        $         +NOSE UP, 500/DEG
006745  120753            ENTAL     INSPHI
006746  163152            SUBAL     PTRPPH                          AL:=AL-PTRPPH(PERTURBATION PHI)
006747  443203            STRAL     PPHI                            PPHI IS RCLL,+RWD,500/DEG
006750  120766            ENTAL     INSPSI
006751  443204            STRAL     PPSI                            PPSI IS HEADING, MAGNETIC,500/DEG
006752  120743            ENTAL     PITR/G
006753  443205            STRAL     PPITRT                          PITCH RATE,+ NOSE UP,500/(DEG/SEC)
006754  120744            ENTAL     RCLR/G
006755  443206            STRAL     PROLRT                          ROLL RATE,+ RWD,500/(DEG/SEC)
006756  120745            ENTAL     YAWR/G
006757  443207            STRAL     PYAWRT                          YAW RATE,+RIGHT,500/(DEG/SEC)
006760  120747            ENTAL     ACCLNG
006761  443210            STRAL     PAX                             LONG. ACCEL.,NO GRAV,+FORW,128/(FT/SEC**2)
006762  120750            ENTAL     ACCLAT
006763  443211            STRAL     PAY                             LAT. ACCEL.,NO GRAV,+RIGHT128/(FT/SEC**2)
006764  120751            ENTAL     ACCNRM
006765  443212            STRAL     PAZ                             NORMAL ACCEL.,NO GRAV,+UP,128/(FT/SEC**2)
006766  120752            ENTAL     AIRSPD
006767  163153            SUBAL     PTRPAS                          AL:=AIRSPC-TRIM AIRSPEED
006770  443213            STRAL     PUVEL                           PUVEL IS PERTURBATION VELOCITY,
                        $         +FORW,8/(FT/SEC)
006771  120765            ENTAL     IVSI                            VERT. SPEED,+UP,32/(FT/SEC)
006772  163154            SUBAL     PTRPHD                          SUBTRACT HDOT AT ENGAGE
006773  443214            STRAL     PWVEL                           PERTURBED VERT. SPEED,+UP,32/(FT/SEC)
006774  120754            ENTAL     BARC
006775  443215            STRAL     PBALT                           BARO. ALT., 16/FT
006776  121147            ENTAL     XPOCT
006777  443216            STRAL     PXBCD                           X POSITION(+ FORW., 16/FT) IN HEADING FRAME
007000  121150            ENTAL     YPOCT
007001  443217            STRAL     PYBCD                           Y POSITION(+ RIGHT, 16/FT) IN HEADING FRAME
007002  121136            ENTAL     ZFILT
007003  443220            STRAL     PZBCD                           Z POSITION(+ DOWN, 16/FT) IN HEADING FRAME
007004  121151            ENTAL     XPDCT
007005  443221            STRAL     PXBDOT                          X INERTIAL VEL.(+FORW, 32/FPS) IN HEADING FRA
                                                                    ME
007006  121152            ENTAL     YPDCT
007007  443222            STRAL     PYBDOT                          Y INERTIAL VEL.(+RIGHT, 32/FPS) IN HEADING FR
                                                                    AME
007010  121142            ENTAL     ZDOT
007011  443223            STRAL     PZBDOT                          Z INERTIAL VEL.(+DOWN, 32/FPS) IN HEADING FRA
                                                                    ME
007012  121134            ENTAL     XFILT                           X POSITION(+ FORW, 16/FT) RUNWAY AXIS
007013  443224            STRAL     PX
007014  121135            ENTAL     YFILT                           Y POSITION(+RIGHT, 16/FT) RUNWAY AXIS
007015  443225            STRAL     PY
007016  121140            ENTAL     XDOT                            X INERTIAL VELOCITY(+ FORW, 32/FPS)RUNWAY AXIS
007017  443226            STRAL     PXDCT
007020  121141            ENTAL     YDOT                            Y INERTIAL VELOCITY(+ RIGHT, 32/FPS)RUNWAY AX
                                                                    IS
007021  443227            STRAL     PYDCT
```

```
1819A ASSEMBLER, VERSION 1.0   NOVR43  10/02/84   SECTION   1                    PAGE   89

 4346                    *
 4347                    *    THIS SECTION DOES THE PYS SELECTION
 4348                    *
 4349  007022  307201              ENTICR   1                                    USE ICR1 AS INDEX I
 4350  007023  360001              ENIBK    1                                    START WITH ICR1=1 SO (I=2)
 4351  007024  133124   PSTP01     ENTALB   PYSSEL-1                             AL:=(PYSSEL+ICR1-1)
 4352  007025  441650              STRAL    TPP                                  TMP1=PYSSEL(1)
 4353  007026  307202              ENTICR   2                                    USE ICR2 AS INDEX
 4354  007027  321650              ENTB     TPP                                  B:=TMP=PYSSEL(I)
 4355  007030  133201              ENTALB   PDTHET-1                            AL:=MEAS(PYSSEL(I))=(PDTHET+TMP-1)
 4356  007031  307201              ENTICR   1                                    USE INDEX REGISTER 1 AGAIN
 4357  007032  452704              STRALB   PYS-1                                (PYS+ICR1-1)=PYS(I)=AL
 4358  007033  566717              PSR      PNMEAS                               IF P=PNMEAS SKIP A LINE
 4359                    *                          ELSE ICR1:=ICR1+1 AND DO NEXT LINE
 4360  007034  347024              JP       PSTP01                               CONTINUE
 4361                    *
 4362                    *
 4363                    *       THIS SECION PUTS EXPERIMENTAL PILOT COMMANDS INTO THE PPLT VECTOR
 4364                    *
 4365  007035  120755   PSTP03     ENTAL    EPLCN                                AL:=EP LONG STICK
 4366  007036  504600              LRTAL    0                                    SCALE THE INPUT OF LONG CYLIC
 4367  007037  442761              STRAL    PPLT                                 PPLT(1)=LONG STICK,1024PER IN,+AFT
 4368  007040  120760              ENTAL    EPMEAV                               AL:=EP COLLECTIVE
 4369  007041  504600              LRTAL    0                                    SCALE THE INPUT OF COLLECTIVE
 4370  007042  442762              STRAL    PPLT+1                               PPLT(2)=COLLECTIVE,1024 PER IN,+UP
 4371  007043  120756              ENTAL    EPLAT                                AL:=EP LATERAL STICK
 4372  007044  504600              LRTAL    0                                    SCALE THE LATERAL STICK
 4373  007045  442763              STRAL    PPLT+2                               PPLT(3)=LATERAL CYCLIC,1024 PER IN,+RIGHT
 4374  007046  120757              ENTAL    EPYAW                                AL:=EP YAW COMMANDS
 4375  007047  504600              LRTAL    0                                    SCALE THE PEDALS
 4376  007050  442764              STRAL    PPLT+3                               PPLT(4)=YAW CONTROL,1024 PER IN,+RIGHT??
 4377                    *
 4378                    *
 4379                    *          ----------SECTION PSTP1----------
 4380                    *
 4381                    *       THIS SECTION DOES THE PID CONTROL CALCULATIONS IN EACH
 4382                    *       OF THE 3 AXES.  IT FIRST CHECKS TO SEE IF THE APPROPRIATE
 4383                    *       PILOT CONTROL IS OUT OF DETENT.  IF OUT OF DETENT, THE PID
 4384                    *       CALCULATIONS ARE HELD AND THE PILOT'S CONTROL BECOMES A
 4385                    *       VELOCITY COMMAND TO THE CH-47.  IF IN DETENT, THE PID CALCULATIONS
 4386                    *       HOLD POSITION IN THAT AXIS.  THE 3 AXES ARE HANDLED INDEPENDENTLY.
 4387                    *       THIS CODE ALSO PUTS A DEADZONE AROUND THE PILOT CONTROL TO
 4388                    *       ELIMINATE TRANSIENTS WHEN GOING IN AND OUT OF DETENT.
 4389                    *
 4390                    *       THE FOLLOWING IS AN EQUIVALENT SET OF BASIC LANGUAGE
 4391                    *       INSTRUCTIONS NEEDED TO DO THE JOB.
 4392                    *
 4393                    *  PSTP10     IF ABS(PPLT(I))<PXDTNT THEN GO TO PSTP11
 4394                    *             ELSE CONTINUE
 4395                    *             PXREF=PIFILT
 4396                    *             PXDOLD=PXEPID
 4397                    *             IF PPLT(I)<0 THEN PPLT(I)=PPLT(I)+PXDTNT
 4398                    *             ELSE PPLT(I)=PPLT(I)-PXDTNT
 4399                    *             GO TO PSTP12
 4400                    *  PSTP11     PDELX=PIFILT-PXREF
 4401                    *             PXDP=PXKP*PDELX
 4402                    *             PXDI=PXDI+PXXI*PDELX
```

Hover Controller Subroutine. (contd)

```
1019A ASSEMBLER, VERSION 1.0   HOVR43  10/02/84   SECTION   1                                        PAGE  90

                   *         PXDD=PKXD*PXDOT
                   *         PXDPID=PXPP+PXKI+PXDD+PXDCLD
                   *
                   *   PSTP12  IDENTICAL EXCEPT FOR THE Y AXIS(PPLT(3) IS LAT. CONTROL)
                   *
                   *   PSTP14 IDENTICAL EXCEPT FOR THE Z AXIS(PPLT(2) IS COLLECTIVE)
                   *
                   *
                   *    X AXIS ***VELOCITY COMMAND/POSITION HOLD*** LOGIC
                   *
007051  122761  PSTP10  ENTAL    PPLT                  AL:=PPLT(1)=PILOT LONG STICK
007052  700333          RJP      ABSVAL                GET ABS(PPLT(1))
007053  023172          CMAL     PXDINT                COMPARE PPLT(1) AND DETENT MAGNITUDE
007054  677115          JPMGR    PSTP11                IF DETENT IS LARGER GO TO PID CALCULATIONS
                   *                       ELSE THE PILOT IS COMMANDING A VELOCITY AND
                   *                       WE DO THE FOLLOWING INSTRUCTIONS
007055  400605          STRZ     W/RFLG                TURN OFF THE PITCH DETENT LIGHT
007056  703777          ENTALK   3777
007057  440111          STRAL    OPR16                 PITCH BAR OUT OF VIEW IF NOT PID
007060  123210          ENTAL    PXBCD                 THESE TWO LINES FORCE THE REFERENCE POSITION
                                                       TO FOLLOW
007061  443231          STRAL    PXBREF                ACTUAL POSITION WHILE THE PILOT IS COMMANDING
                                                        A RATE.
007062  123231          ENTAL    PXBREF                DO THE TRANSFORMATION FROM HEADING TO RUNWAY
007063  103242          ENTAU    PYBREF                GET XBREF AND YBREF IN THE A REGISTER
007064  703413          RJP      PROTAT
007065  443230          STRAL    PXREF
007066  403241          STRAU    PYREF
                   *
                   *      DO FADE OUT OF OLD PID CONTROL WHILE PILOT IS
                   *      COMMANDING A VELOCITY.
007067  123271          ENTAL    PXBCFF                AL:=FADE-OUT MULTIPLIER
007070  637074          JPALNZ   LCK+4                 IF NOT RAMPED TO ZERO, GO DOWN AND CONTINUE R
                                                       AMP
007071  403240          STRZ     PXDCLD                IF WE'VE RAMPED TO ZERO, HOLD PXDCLD AT ZERO
007072  403271          STRZ     PXBCFF                IF WE'VE RAMPED TO ZERO, KEEP RAMP AT ZERO
007073  347101          JP       LCK+6                 SKIP AROUND THE FADE-OUT TO STRZ' COMMAND
007074  717770          ADDALK   -1                    CONTINUE RAMP DOWN
007075  443271          STRAL    PXBCFF                STORE THE RAMPED DOWN VALUE
007076  243240          MULAL    PXDCLD                A:=RAMP*PXDOLC
007077  504706          LRTA     6C                    DIVIDE BY NUMBER OF CYCLES IN RAMP DOWN
007100  403240          STRAU    PXDCLD                PXDOLCD= FADED VALUE
007101  403266          STRZ     PXEZON                GET READY FOR PID FADE-IN
007102  123240          ENTAL    PXDCLD                STORE THE FADED VALUE OF THE LAST PID
007103  443274          STRAL    PXCLST                COMMAND BEFORE COMING OUT OF DETENT
007104  403260          STRZ     PXDI                  ZERO THE INTEGRATOR
007105  403261          STRZ     PXDI+1                *
                   *
                   *   NOW DO THE DEADZONE CALCULATIONS
007106  122761          ENTAL    PPLT                  AL:=PPLT(1), PILOT LNG STICK
007107  677112          JPALNG   LOK+3                 IF CONTROL NEGATIVE, SKIP A LINE
007110  163172          SUBAL    PXDINT                ELSE THE CONTROL IS POSITIVE SO SUBTRACT
                   *                       THE DETENT VALUE.
007111  347113          JP       LCK+2                 GO STORE THE DEADZONED VALUE
007112  143172          ADDAL    PXDINT                FOR NEG.CONTROL, ADD DETENT
007113  442761          STRAL    PPLT                  PUT THE MODIFIED CONTROL BACK IN PPLT
007114  347205          JP       PSTP12                GO TO THE Y AXIS
```

```
1819X ASSEMBLER, VERSION 1.0   HOVP43  10/02/84   SECTION   1                                   PAGE   91

 4457  007115  123230   PSTP11   ENTAL    PXREF                  DO THE TRANSFORMATION FROM RUNWAY TO HEADING
 4458  007116  103261            ENTAU    PYREF                  GET XREF  AND YREF  IN THE A REGISTER
 4459  007117  763366            RJP      ACTATE
 4460  007120  483262            STRAU    PYBREF
 4461  007121  700001            ENTALK   1                      TURN ON THE  PITCH DETENT LIGHT
 4462  007122  480685            STRAL    M/MFLG
 4463  007123  402761            STRZ     PPLT                   STORE ZERO IN PILOT COMMAND WHEN IN PID
 4464  007124  123216            ENTAL    PXBCD                  THESE 3 LINES CALCULATE
 4465  007125  163231            SUBAL    PXBREF                 THE POSITION ERROR, AL=PDELX
 4466  007126  483232            STRAL    PDELX                  STORE PDELX
 4467  007127  506100            CPAL                            USE PITCH BAR TO SHOW XPOSN ERROR
 4468  007130  760823            RJP      SATCHK                 IN PID PCDE
 4469  007131  440111            STRAL    QPR16                  SCALING 3.33V=602 COUNTS=42.6 FEET=1 INCH?
 4470  007132  123232            ENTAL    PDELX
 4471  007133  243155            PULAL    PKXP                   MULTIPLY PKXP*PDELX(DO PROPORTIONAL CONTROL)
 4472  007134  504707            LRTA     7C                     SCALING ON KXP IS 1 FPS PER FT ERROR = 1024 B
                                                                  ITS
 4473  007135  483233            STRAU    PXDF                   STORE THE RESULT
 4474  007136  123232            ENTAL    PDELX                  GET POSITION ERROR TO BE INTEGRATED(1FT=16BIT
                                                                  S)
 4475  007137  760533            RJP      ABSVAL                 HOLD INTEGRATOR TERM IF ERROR GT 15FT
 4476  007140  023176            CMAL     PK240
 4477  007141  657147            JPPLEQ   PS3115
 4478  007142  123232            ENTAL    PDELX
 4479  007143  243160            PULAL    PKXI                   MULTIPLY BY DIGITIZED INTEGRAL GAIN(1FPS/FT=2
                                                                  **19BITS)
 4480  007144  203260            ADDA     PXDI                   DO SUMMATION(THE INTEGRATION)
 4481  007145  500600            STRA     PXDI                   AU GOES TO PXDI+1, AL TO PXDI
       007146  033260
 4482  007147  123226   PS3115   ENTAL    PXDCT                  DO DERIVATIVE CONTROL
 4483  007150  243163            PULAL    PKXD                   PXDD=PKXD*PXDOT
 4484  007151  504700            LRTA     6C                     SCALING ON PKXD IS 1 FPS PER FPS * 1024 BITS
 4485  007152  463234            STRAU    PXDE                   STORE RESULT
 4486  007153  123202            ENTAL    PCTHET                 GET THETA FOR 2ND DERIVATIVE CONTROL
 4487  007154  243166            PULAL    PKXTH                  PXDTH=PCTHET*PKXTH
 4488  007155  504704            LRTA     4C                     SCALING ON PKXTH IS 1 FPS PER DEG * 1048 BITS
 4489  007156  463235            STRAU    PXDTH                  STORE THE RESULT
 4490  007157  123205            ENTAL    PPITRT                 GET Q FOR 3RD DERIVATIVE CONTROL
 4491  007160  243170            PULAL    PKXPR                  PXDPR=PPITRT*PKXPR
 4492  007161  504704            LRTA     4C                     SCALING ON PKXPR IS 1 FPS PER DEG/SEC * 1048
                                                                  BITS
 4493  007162  483236            STRAU    PXDPR                  STORE THE RESULT
 4494  007163  123233            ENTAL    PXDF                   BEGIN THE PXDPID EQUATION
 4495  007164  143201            ADDAL    PXDI+1
 4496  007165  143234            ADDAL    PXDE
 4497  007166  143235            ADDAL    PXDTH
 4498  007167  143236            ADDAL    PXDPR
 4499  007170  483237            STRAL    PXDPID                 PXDPID=PXDP+PXDI+PXDD+PXDTH+PXDPR
 4500                  *
 4501                  *       DO RAMP FADE-IN OF PID COMMAND WHEN THE PILOT
 4502                  *       GOES BACK INTO DETENT
 4503  007171  123208            ENTAL    PXFZCN                 ACIM FADER MULTIPLIER
 4504  007172  710001            ADDALK   1                      DO RAMP UP
 4505  007173  103201            ENTAU    PK128                  USE 128 CYCLES AS THE RAMP UP TIME
 4506  007174  760537            RJP      LIMIR                  LIMIT PXFZCN TO PK128 (LIMIT AL TC AU)
 4507  007175  483208            STRAL    PXFZCN                 STORE RAMP FADER VALUE
 4508  007176  243237            PULAL    PXDPID                 A:=RAMP*PID CONTROL
```

Hover Controller Subroutine. (contd)

```
1019A ASSEMBLER, VERSION 1.0   HOVR43  10/02/84   SECTION   1                                   PAGE   97

007177  504307           RSHA     7C          DIVIDE BY NUMBER OF CYCLES IN RAMP
007200  143274           ADDAL    PXDIST      ADD LAST PID COMMAND(FADED) TO AVCIC TRANSIENT
007201  443237           STRAL    PXDFID      PUT SUM BACK INTO PID COMMAND
007202  443240           STRAL    PXDCLD      STORE PID COMMAND FOR LATER FADE=OUT
007203  123200           ENTAL    PK64        GET READY FOR THE PXDOLD FADE-OUT ABOVE
007204  443271           STRAL    PXECFF      CXECFF GET START OF FADE=OUT VALUE
                 *
                 *    Y AXIS ***VELOCITY COMMAND/POSITION HOLD***
                 *
007205  122763  PSTP12   ENTAL    PPLT+2      AL:=PPLT(3)=PILOT LAT STICK
007206  760533           RJP      ABSVAL      GET ABS(PPLT(3))
007207  023173           CMAL     PYDTNT      COMPARE PPLT(3) AND DETENT MAGNITUDE
007210  677251           JPMGR    PSTP13      IF DETENT IS LARGER GO TO PID CALCULATIONS
                 *                        ELSE THE PILOT IS COMMANDING A VELOCITY AND
                 *                        WE DO THE FOLLOWING INSTRUCTIONS
007211  400004           STRZ     R/PFLG      TURN OFF THE ROLL DETENT LIGHT
007212  703777           ENTALK   3777        ROLL BAR OUT OF VIEW IF NOT IN PID
007213  440112           STRAL    OPR17
007214  123217           ENTAL    PYBCD       THESE TWO LINES FORCE THE REFERENCE POSITION
                                              TO FOLLOW
007215  443242           STRAL    PYBREF      ACTUAL POSITION WHILE THE PILOT IS COMMANDING
                                               A RATE.
007216  123231           ENTAL    PXBREF      DO THE TRANSFORMATION FROM HEADING TO RUNWAY
007217  103242           ENTAU    PYBREF      GET XRREF AND YRREF IN THE A REGISTER
007220  763413           RJP      BROTAT
007221  443230           STRAL    PXREF
007222  463241           STRAU    PYREF
                 *
                 *    DO FADE OUT OF OLD PID CONTROL WHILE PILOT IS
                 *    COMMANDING A VELOCITY.
007223  123272           ENTAL    PYECFF      AL:=FADE-OUT MULTIPLIER
007224  037230           JPALNZ   LCK+4       IF NOT RAMPED TO ZERO, GO DOWN AND CONTINUE R
                                              AMP
007225  403251           STRZ     PYDCLD      IF WE'VE RAMPED TO ZERO, HOLD PYDOLD AT ZERO
007226  403272           STRZ     PYECFF      IF WE'VE RAMPED TO ZERO, KEEP RAMP AT ZERO
007227  347235           JP       LCK+6       SKIP AROUND THE FADE-OUT TO STRZ' COMMAND
007230  717776           ADDALK   -1          CONTINUE RAMP DOWN
007231  443272           STRAL    PYECFF      STORE THE RAMPED DOWN VALUE
007232  243251           MULAL    PYDCLD      A:=RAMP*PYDOLD
007233  504706           RSHA     6C          DIVIDE BY NUMBER OF CYCLES IN RAMP DOWN
007234  463251           STRAU    PYDCLD      PYDOLD:= FADED VALUE
007235  403267           STRZ     PYEZON      GET READY FOR PID FADE=IN
007236  123251           ENTAL    PYDCLD      STORE THE FADED VALUE OF THE LAST PID
007237  443275           STRAL    PYDIST      COMMAND BEFORE COMING OUT OF DETENT
007240  403262           STRZ     PYCI        ZERO THE INTEGRATOR
007241  403263           STRZ     PYCI+1      *
                 *
                 *    NOW DO THE DEADZONE CALCULATIONS
007242  122763           ENTAL    PPLT+2      AL:=PPLT(3), PILOT LATERAL STICK
007243  077240           JPALNG   LCK+3       IF CONTROL NEGATIVE, SKIP A LINE
007244  163173           SUBAL    PYDTNT      ELSE THE CONTROL IS POSITIVE SO SUBTRACT
                 *                        THE DETENT VALUE.
007245  347247           JP       LCK+2       GO STORE THE DEADZONED VALUE
007246  143173           ADDAL    PYDTNT      FOR NEG.CONTROL, ADD DETENT
007247  442763           STRAL    PPLT+2      PUT THE MODIFIED CONTROL BACK IN PPLT
007250  347335           JP       PSTP14      GO TO THE Z AXIS
```

Hover Controller Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   HCVR43  10/02/84   SECTION  1                                              PAGE  93

007251  123230  PSTP13  ENTAL    PXREF      DO THE TRANSFORMATION FROM RUNWAY TO HEADING
007252  103281          ENTAU    PYREF      GET XREF  AND YREF  IN THE A REGISTER
007253  763366          RJP      ROTATE
007254  443231          STRAL    PXHREF
007255  700001          ENTALK   1          TURN ON THE ROLL DETENT LIGHT
007256  440664          STRAL    A/PFLG
007257  402763          STRZ     PFL1+2     STORE ZERO IN THE PILOT COMMAND WHEN IN PID
007260  123217          ENTAL    PYBCC      THESE 3 LINES CALCULATE
007261  163242          SUBAL    PYHREF     THE POSITIN ERROR, AL=PDELY
007262  443243          STRAL    PDELY      STORE PDELY
007263  506100          CPAL                SHOW Y POSN ERROR ON ROLL BAR
007264  760623          RJP      BSYCHK     IF IN PIC
007265  440112          STRAL    OPR17      SCALING 3.33V=682 COUNTS=42.6 FEET=1 INCH?
007266  123243          ENTAL    PDELY
007267  243156          MULAL    PKYP       MULTIPLY PKYP*PDELY(DO PROPORTIONAL CONTROL)
007270  504707          LRTA     7C         SCALING ON KYP IS 1 FPS PER FT ERROR = 1024 B
                                            ITS
007271  463244          STRAU    PYDP       STORE THE RESULT
007272  123243          ENTAL    PDELY      GET POSITION ERROR TO BE INTEGRATED(1FT=16BIT
                                            S)
007273  243161          MULAL    PKYI       MULTIPLY BY DIGITIZED INTEGRAL GAIN(1FPS/FT=2
                                            **19BITS)
007274  203262          ADDA     PYDI       DO SUMMATION(THE INTEGRATION)
007275  500600          STRA     PYDI       AU GOES TO PYKI+1, AL TO PYKI
007276  033262
007277  123227          ENTAL    PYDOT      DO DERIVATIVE CONTROL
007300  243164          MULAL    PKYD       PYDD=PKYD*PYDOT
007301  504706          LRTA     6C         SCALING ON PKYD IS 4 FPS PER FPS = 4096 BITS
007302  463245          STRAU    PYDD       STORE RESULT
007303  123203          ENTAL    PPHI       GET PHI(ROLL ANGLE) FOR 2ND DERIVATIVE CONTROL
007304  243167          MULAL    PKYPH      PYDPH=PPHI*PKYPH
007305  504704          LRTA     4C         SCALING ON PKYPH IS 1 FPS PER DEG = 1048 BITS
007306  463246          STRAU    PYDPH      STORE THE RESULT
007307  123208          ENTAL    PROLRT     GET P FOR 3RD DERIVATIVE CONTROL
007310  243171          MULAL    PKYRR      PYDRR=PROLRT*PKYRR
007311  504704          LRTA     4C         SCALING ON PKYRR IS 1FPS PER DEG/SEC = 1048 B
                                            ITS
007312  463247          STRAU    PYDRR      STORE THE RESULT
007313  123244          ENTAL    PYDP       BEGIN THE PYDPIC EQUATION
007314  143263          ADDAL    PYDI+1
007315  143245          ADDAL    PYDD
007316  143246          ADDAL    PYDPH
007317  143247          ADDAL    PYDRR
007320  443250          STRAL    PYDPID     PYDPIC=PYDP+PYDI+PYDD+PYDPH+PYDRR
                *
                *       DO RAMP FADE-IN OF PID COMMAND WHEN THE PILOT
                *       GOES BACK INTO DETENT
007321  123207          ENTAL    PYEZCN     ALIGN FADER MULTIPLIER
007322  710001          ADDALK   1          DO RAMP UP
007323  103201          ENTAU    PK528      USE 128 CYCLES AS THE RAMP UP TIME
007324  760537          RJP      LIMIR      LIMIT PYEZCN TO PK128 (LIMIT AL TO AU)
007325  443207          STRAL    PYEZCN     STORE RAMP FADER VALUE
007326  243250          MULAL    PYDPID     AI=RAMP*PID CONTROL
007327  504307          RSHA     7C         DIVIDE BY NUMBER OF CYCLES IN RAMP
007330  143275          ADDAL    PYDLST     ADD LAST PID COMMAND(FADED) TO AVOID TRANSIENT
```

Hover Controller Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   HOVR43  10/02/84   SECTION   1                                         PAGE   94

007331  443250              STRAL    PYDFID              PUT SUM BACK INTO PID COMMAND
007332  443251              STRAL    PYDCLD              STORE PID COMMAND FOR LATER FADE-OUT
007333  123200              ENTAL    PK64                GET READY FOR THE PYDOLD FADE-OUT ABOVE
007334  443272              STRAL    PYECFF              PYECFF GET START OF FADE-OUT VALUE
                *
                *   Z AXIS ***VELOCITY COMMAND/POSITION HOLD***
                *
007335  122762  P8TP14      ENTAL    PPLT+1              AL=PPLT(2)=PILOT COLLECTIVE
007336  760533              RJP      ABSVAL              GET ABS(PPLT(2))
007337  023174              CMAL     PZDTNT              COMPARE PPLT(2) AND DETENT MAGNITUDE
007340  677372              JPMGR    P8TF15              IF DETENT IS LARGER GO TO PID CALCULATIONS
                *                       ELSE THE PILOT IS COMMANDING A VELOCITY AND
                *                       WE DO THE FOLLOWING INSTRUCTIONS
007341  000003              STRZ     G/NFLG              TURN OFF THE COLLECTIVE DETENT LIGHT
007342  123220              ENTAL    PZBCD               THESE TWO LINE FORCE THE REFERENCE POSITION T
                                                         O FOLLOW
007343  443252              STRAL    PZREF               ACTUAL POSITION WHILE THE PILOT IS COMMANDING
                                                          A RATE.
                *
                *   DO FADE OUT OF OLD PID CONTROL WHILE PILOT IS
                *   COMMANDING A VELOCITY.
007344  123273              ENTAL    PZECFF              AL=FADE-OUT MULTIPLIER
007345  637351              JPALNZ   LCK+4               IF NOT RAMPED TO ZERO, GO DOWN AND CONTINUE R
                                                         AMP
007346  403257              STRZ     PZDCLD              IF WE'VE RAMPED TO ZERO, HOLD PZDCLD AT ZERO
007347  403273              STRZ     PZECFF              IF WE'VE RAMPED TO ZERO, KEEP RAMP AT ZERO
007350  347356              JP       LCK+6               SKIP AROUND THE FADE-OUT TO STRZ' COMMAND
007351  717776              ADDALK   -1                  CONTINUE RAMP DOWN
007352  443273              STRAL    PZECFF              STORE THE RAMPED DOWN VALUE
007353  243257              MULAL    PZDCLD              AI=RAMP*PZDOLD
007354  504706              LRIA     6C                  DIVIDE BY NUMBER OF CYCLES IN RAMP DOWN
007355  463257              STRAU    PZDCLD              PZDOLD=FADED VALUE
007356  403270              STRZ     PZEZON              GET READY FOR PID FADE-IN
007357  123257              ENTAL    PZDCLD              STORE THE FADED VALUE OF THE LAST PID
007360  443276              STRAL    PZCLST              COMMAND BEFORE COMING OUT OF DETENT
007361  403264              STRZ     PZCI                ZERO THE INTEGRATOR
007362  403265              STRZ     PZCI+1              *
                *
                *   NOW DO THE DEADZONE CALCULATIONS
007363  122762              ENTAL    PPLT+1              AL=PPLT(2), PILOT COLLECTIVE
007364  677367              JPALNG   LCK+3               IF CONTROL NEGATIVE, SKIP A LINE
007365  163174              SUBAL    PZDTNT              ELSE THE CONTROL IS POSITIVE SO SUBTRACT
                *                       THE DETENT VALUE.
007366  347370              JP       LCK+2               GO STORE THE DEADZONED PILOT COMMAND
007367  143174              ADDAL    PZDTNT              FOR NEG.CONTROL, ADD DETENT
007370  442762              STRAL    PPLT+1              PUT THE MODIFIED CONTROL BACK IN PPLT(2)
007371  347434              JP       P8TF2               GO TO THE PILOT SCALING ROUTINE
007372  402762  P8TP15      STRZ     PPLT+1              STORE ZERO IN THE PILOT COMMAND WHEN IN PID
007373  700001              ENTALK   1                   TURN ON THE COLLECTIVE DETENT LIGHT
007374  440003              STRAL    G/NFLG
007375  123220              ENTAL    PZBCD               THESE 3 LINES CALCULATE
007376  163252              SUBAL    PZREF               THE POSITION ERROR, AL=PDELZ
007377  443253              STRAL    PDELZ               STORE PDELZ
007400  243157              MULAL    PKZP                MULTIPLY PKZP*PDELZ(DO PROPORTIONAL CONTROL)
007401  504707              LRIA     7C                  SCALING ON KZP IS 1 FPS PER FT ERROR = 1024 B
                                                         ITS
007402  463254              STRAU    PZDP                STORE THE RESULT
```

Hover Controller Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   HOVR43  10/02/84   SECTION   1                                   PAGE   95

007403 123253         ENTAL    PCELZ          GET POSITION ERROR TO BE INTEGRATED(1FT=16BIT
                                              S)
007404 243162         MULAL    PKZI           MULTIPLY BY DIGITIZED INTEGRAL GAIN(1FPS/FT=2
                                              **19BITS)
007405 203264         ADDA     PZCI           DO SUMMATION(THE INTEGRATION)
007406 500600         STRA     PZCI           AU GOES TO PZKI+1, AL TO PZKI
007407 033264
007410 123223         ENTAL    PZBDOT         DO DERIVATIVE CONTROL
007411 243165         MULAL    PKZD           PZDD=PKZD*PZBDOT
007412 504706         LRTA     6C             SCALING ON PKZD IS 1 FPS PER FPS = 1024 BITS
007413 443255         STRAU    PZDC           STORE RESULT
007414 123254         ENTAL    PZDF           BEGIN THE PZDPIC EQUATION
007415 143265         ADDAL    PZCI+1
007416 143255         ADDAL    PZDC
007417 443256         STRAL    PZDFID         PZDPIC=PZCP+PZDI+PZDD
                *
                *   DO RAMP FADE-IN OF PID COMMAND WHEN THE PILOT
                *   GOES BACK INTO DETENT
007420 123270         ENTAL    PZEZCN         AL1= FACER MULTIPLIER
007421 710001         ADDALK   1              DO RAMP UP
007422 103201         ENTAU    PK128          USE 128 CYCLES AS THE RAMP UP TIME
007423 760537         MJP      LIMIN          LIMIT PZEZCN TO PK64 (LIMIT AL TO AU)
007424 443270         STRAL    PZEZCN         STORE RAMP FACER VALUE
007425 243256         MULAL    PZDFID         A1=RAMP*PID CONTROL
007426 504307         RSHA     7C             DIVIDE BY NUMBER OF CYCLES IN RAMP
007427 143276         ADDAL    PZCLST         ADD LAST PID COMMAND(FADEC) TO AVOID TRANSIENT
007430 443256         STRAL    PZDFID         PUT SUM BACK INTO PID COMMAND
007431 443257         STRAL    PZDCLD         STORE PID COMMAND FOR LATER FADE-OUT
007432 123200         ENTAL    PK64           GET READY FOR THE PZDOLD FADE-OUT ABOVE
007433 443273         STRAL    PZECFF         PZEOFF GET START OF FADE-OUT VALUE
                *
                *   --------SECTION PSTP2--------------
                *
                *   THIS SECTION PUTS (PPLT * COMMAND SCALING) INTO THE PYDES
                *   (DESIRED OUTPUT) VECTOR.
                *
                *          PYDES = PYSCLE * PPLT
                *
                *    WHERE PYSCLE IS THE DIAGONAL SCALING MATRIX
007434 507310  PSTP2  ENTSR    1C             USE MEMORY BANK 0
007435 507201         ENTICR   1              USE ICR1 AS THE INDEX REG.(B)
007436 300000         ENTBK    0              INITIALIZE THE B REG TO 0
007437 507313         ENTSR    13             USE MEMORY BANK 3
007440 132761  PSTP20 ENTALB   PPLT           AL1=(PPLT+ICR1)
007441 253003         MULALB   PYSCLE         A1=AL*(PYSCLE + ICR1)
007442 504702         LRTA     2              SCALING WAS 2**16 FOR PSTP2
007443 472777         STRAUB   PYDES          STORE RESULT IN (PYDES+ICR1),SCALING IS 2**16
007444 306715         BSK      PNCP1          IF ICR1=PNCR1(NC-1) THEN SKIP NEXT LINE
                *                     ELSE DO NEXT LINE AND ICR1=ICR1+1
007445 347440         JP       PSTP20         GO SCALE THE NEXT CONTROL
007446 347447         JP       PSTP3          GO TO THE INTEGRAL CONTROL SECTION
                *
                *   --------SECTION PSTP3----------
                *
                *   FIRST DO THE SET POINT GENERATOR MATRIX MULTIPLICATION
                *                      PUC = PNK * PYDES
```

Hover Controller Subroutine. (contd)

```
                                                                                           PAGE  96
                    $              WHERE PUC = PNC X PNC   AND  PYDES = PNC X 1
                    $
                    $    PYDES INCLUDES BOTH PILOT COMMANDS AND THE
                    $    PID CONTROLLER COMMANDS.
                    $
007447  507310  PSTP3   ENTSR    $C                         USE MEMORY BANK 0
007450  507201          ENTICR   1                          USE ICR1 AS THE INDEX REG.(B)
007451  360000          ENTBK    0                          INITIALIZE THE B REG. TO 0
007452  400002          STRZ     2                          ICR2=0, THE COLUMN INDEX
007453  400003          STRZ     3                          ICR3=0, THE ROW INDEX
007454  507313          ENTSR    13                         USE MEMORY BANK 3
007455  123237          ENTAL    PXDPID                     AL:=PXDPID
007456  142777          ADDAL    PYDES                      PYDES(1)=PYDES(1)+PXDPID
007457  442777          STRAL    PYDES                      STORE RESULT
007460  123256          ENTAL    PZDPID                     AL:=PZDPID
007461  143000          ADDAL    PYDES+1                    AL:=PYDES(2)+PYDPID, LATERAL
007462  443000          STRAL    PYDES+1                    STORE RESULT IN PYDES(2)
007463  123250          ENTAL    PYDPID                     AL:=PYDPID
007464  143001          ADDAL    PYDES+2                    AL:=PYDES(3)+PZDPID
007465  443001          STRAL    PYDES+2                    STORE IN PYDES(3), COLLECTIVE
007466  123136          ENTAL    PNC                        AL:=NUMBER OF CONTROLS
007467  717776          ADDALK   =1                         AL:=PNC=1
007470  507310          ENTSR    $C                         USE MEMORY BANK 0
007471  446711          STRAL    PRWEND                     PRWEND:=PNC=1
007472  507313          ENTSR    13                         USE MEMORY BANK 3
007473  700000  PNXRW1  ENTALK   0                          AL:=0
007474  507400          XFRLU                               AU:=AL=0
007475  500600  PNXCL1  STRA     TMP                        TMP:=A
007476  031650
007477  507202          ENTICR   2                          GET THE COLUMN INDEX
007500  370001          ENTBKP   1                          INCREMENT COLUMN INDEX(ICR2=ICR2+1)
007501  132776          ENTALB   PYDES-1                    AL:=(PYDES+ICR2-1)
007502  507201          ENTICR   1                          USE ICR1 AS INDEX REG.(ELEMENT INDEX)
007503  253000          MULALB   PNR                        A:=AL * (PNR+ICR1)
007504  201650          ADDA     TMP                        A:=A + (TMP,TMP+1)
007505  505300          SKPNOV                              CHECK FOR OVERFLOW
007506  504000          NOOP                                CALL OVERFLOW HANDLER
007507  566711          BSK      PRWEND                     IF ICR1=PRWEND THEN SKIP NEXT LINE
                    $                                       ELSE DO NEXT LINE AND ICR1=ICR1+1
007510  347475          JP       PNXCL1                     JUMPS TO NEXT COLUMN OF PNR
                    $                                       AND NEXT ROW ELEMENT OF PPLT
007511  370001          ENTBKP   1                          ICR1=ICR1+1
007512  507203          ENTICR   3                          USE ICR3 AS INDEX REG.(ROW INDEX)
007513  504705          LRTA     SC                         SCALING FOR PSTP3,2**13
007514  472745          STRAUB   PUC                        PUC+ICR3:=(AU)
007515  566715          BSK      PNCM1                      IF ICR3=PNCM1 THEN SKIP NEXT LINE
                    $                                       ELSE DO NEXT LINE AND ICR3=ICR3+1
007516  347520          JP       PRSET1                     GO RESET THE COUNTER
007517  347530          JP       P2TF4                      GO TO NEXT MATRIX MULT IF DONE
007520  507202  PRSET1  ENTICR   2                          SAVE ROW INDEX AND START USING COLUMN INDEX
007521  360000          ENTBK    0                          RESET COLUMN INDEX TO ZERO
007522  507310          ENTSR    $0                         BANK 0 TO USE PRWEND AND PNCTRL
007523  126711          ENTAL    PRWEND                     AL:=PRWEND
007524  146716          ADDAL    PNCTRL                     AL:=PRWEND+PNCTRL=PRWEND+# PP COLUMNS
007525  446711          STRAL    PRWEND                     PRWEND:=PRWEND+PNCTRL
007526  507313          ENTSR    13                         BANK 3 FOR DATA
007527  347473          JP       PNXRW1                     DO THE NEXT ROW
```

Hover Controller Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   HOVR43  10/02/84   SECTION   1                                                    PAGE   97

                             *
                             * ================SECTION PSTP4====================
                             *
                             *           THIS IS THE 2ND MATRIX MULTIPLY IN THE CMPSTR
                             *           SUBROUTINE, I.E.
                             *                     PEMIN * PYS
 007530  507310   PSTP4     ENTSR    $C                                      USE MEMORY BANK 0
 007531  507201             ENTICR   1                                       USE ICR1 AS THE INDEX REG.(B)
 007532  360000             ENTBK    0                                       INITIALIZE THE B REG. TO 0
 007533  400002             STRZ     2                                       ICR2=0, THE COLUMN INDEX
 007534  400003             STRZ     3                                       ICR3=0, THE ROW INDEX
 007535  507313             ENTSR    13                                      USE MEMORY BANK 3
 007536  123137             ENTAL    PMM                                     AL:=NUMBER OF MEASUREMENTS
 007537  717776             ADDALK   -1                                      AL:=PMM-1
 007540  507310             ENTSR    $0                                      USE MEMORY BANK 0 SINCE PMMEND IS THERE
 007541  446711             STRAL    PMMEND                                  PMMEND:=PMM-1
 007542  507313             ENTSR    13                                      BACK TO MEMORY BANK 3 FOR DATA
 007543  700000   PNXRW2    ENTALK   0                                       AL:=0
 007544  507400             XFRLU                                            AU:=AL=0
 007545  500600   PNXCL2    STRA     TMP                                     TMP:=A
 007546  031650
 007547  507202             ENTICR   2                                       GET THE COLUMN INDEX
 007550  370001             ENTBKB   1                                       INCREMENT COLUMN INDEX(ICR2=ICR2+1)
 007551  132764             ENTALB   PYS-1                                   AL:=(PYS+ICR2-1)
 007552  507201             ENTICR   1                                       USE ICR1 AS INDEX REG.(ELEMENT INDEX)
 007553  253030             MULALB   PEMIN                                   A:=AL * (PEMIN+ICR1)
 007554  201650             ADDA     TMP                                     A:=A + (TMP,TMP+1)
 007555  505300             SKPNOV                                           CHECK FOR OVERFLOW
 007556  504000             NOOP                                             CALL OVERFLOW HANDLER
 007557  566711             ESK      PMMEND                                  IF ICR1=PMMEND THEN SKIP NEXT LINE
                             *                                ELSE DO NEXT LINE AND ICR1=ICR1+1
 007560  347545             JP       PNXCL2                                  JUMPS TO NEXT COLUMN OF PEMIN
 007561  370001             ENTBKB   1                                       ICR1=ICR1+1
 007562  507203             ENTICR   3                                       USE ICR3 AS INDEX REG.(ROW INDEX)
 007563  504704             LRTA     4C                                      PSTP4 SCALING IS 2**14
 007564  473024             STRAUB   PZKPI                                   PZKPI+ICR3I=(AU)
 007565  566712             ESK      PRRP1                                   IF ICR=PRR THEN SKIP NEXT LINE
                             *                           ELSE DO NEXT LINE AND ICR3=ICR3+1
 007566  347570             JP       PRSET2                                  GO RESET THE COUNTER
 007567  347600             JP       PSTP5                                   GO TO NEXT MATRIX MULT IF DONE
 007570  507202   PRSET2    ENTICR   2                                       SAVE ROW INDEX AND START USING COLUMN INDEX
 007571  360000             ENTBK    0                                       RESET COLUMN INDEX TO ZERO
 007572  507310             ENTSR    $0                                      BANK 0 TO USE PMMEND AND PNMEAS
 007573  126711             ENTAL    PMMEND                                  AL:=PMMEND
 007574  146717             ADDAL    PNMEAS                                  AL:=PMMEND+PNMEAS
 007575  446711             STRAL    PMMEND                                  PMMEND:=PMMEND+PNMEAS=PMMEND+#OF COLUMNS
 007576  507313             ENTSR    13                                      BACK TO MEMORY BANK 3 FOR DATA
 007577  347543             JP       PNXRW2                                  DO THE NEXT ROW
                             *
                             * ================SECTION PSTP5==========
                             *
                             *           THIS STEP(PSTP5) DOES THE MATRIX MULTIFLICATION
                             *           DESCRIBED BY
                             *                  PAMIN*PZK
                             *           WHERE PAMIN IS THE MINIMAL FORM OF THE COMPENSATOR
                             *                DYNAMICS MATRIX
 007600  507202   PSTP5     ENTICR   2                                       SELECT ICR2(J=JCR2)
```

Hover Controller Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION   1                                              PAGE  98

  007601  360000             ENTBK    0                             ICR2:=0
  007602  507201             ENTICR   1                             SELECT ICR1(I=ICR1)
  007603  360000             ENTBK    0                             ICR1:=1
  007604  566713   PSTP50    ESK      PPR                           IF ICR=PR SKIP ONE
                *              LINE ELSE ICR1=ICR1+1(I=I+1) AND     DO NEXT LINE
  007605  347607             JP       PSTP51                        GO CHECK FOR TYPE OF MINIMAL BLOCK
  007606  347645             JP       PSTP54                        GO TO SUMMATION WITH PZKP1
  007607  507202   PSTP51    ENTICR   2                             USE ICR2 AS INDEX,J
  007610  133140             ENTALB   PADESC                        AL:=(PADESC+ICR2)
  007611  023311             CMAL     PCNE                          COMPARE AL,I.E. PADESC(J), WITH 1
  007612  617616             JPEQ     PSTP52                        GOTO PSTP52 IF PADESC(J)=1
  007613  023312             CMAL     PTWC                          COMPARE AL TO 2
  007614  617625             JPEQ     PSTP53                        GOTO PSTP53 IF PADESC(J)=2
  007615  504000             NOOP                                   A CALL TO AN ERROR HANDLER GOES HERE
                *        THIS IS THE SECTION FOR 1X1 BLOCKS OF AMIN
  007616  370001   PSTP52    ENTBKB   1                             ICR2=ICR2+1,I.E. J=J+1
  007617  507201             ENTICR   1                             ICR1 IS THE INDEX REG.
  007620  133030             ENTALB   PAMIN-1                       AL:=AL*(PAMIN+ICR1-1)
  007621  253016             MULALB   PZK-1                         A:=AL*(PZK+ICR1-1)
  007622  504703             LRTA     3                             SCALING FOR PSTP52 IS 2**15
  007623  473276             STRAUB   TMPPZ-1                       TMPPZ=1:=AU=A(I)*PZK(I),SCALING IS 2**15
  007624  347604             JP       PSTP50                        JUMP BACK TO GET NEXT AMIN DESCRIPTOR
                *        THIS IS THE SECTION FOR 2X2 BLOCKS OF AMIN
  007625  370001   PSTP53    ENTBKB   1                             ICR2=ICR2+1,I.E. J=J+1
  007626  507201             ENTICR   1                             ICR1 IS NOW THE INDEX REG.
  007627  133017             ENTALB   PZK                           AL:=(PZK+ICR1)
  007630  453276             STRALB   TMPPZ-1                       TMPPZ+ICR1-1:=(PZK+ICR1),I.E.
                *                         TMPPZ(I)=PZK(I+1)
  007631  133030             ENTALB   PAMIN-1                       AL:=(PAMIN+ICR1-1)
  007632  253016             MULALB   PZK-1                         A:=AL*(PZK+ICR1-1)
  007633  504703             LRTA     3                             SCALING FOR PSTP53 IS 2**15
  007634  461050             STRAU    TMP                           TMP:=AU, I.E. TMP=PAMIN(I)*PZK(I),SCALING IS
                                                                    2**15
  007635  133031             ENTALB   PAMIN                         AL:=(PAMIN+ICR1)
  007636  253017             MULALB   PZK                           A:=AL*(PZK+ICR1),I.E.
  007637  504703             LRTA     3                             SCALING FOR PSTP53 IS 2**15
                *                         A=PAMIN(I+1)*PZK(I+1)
  007640  507500             XPROD                                  PUT AU INTO AL
  007641  141050             ADDAL    TMP                           AL:=AL+TMP,I.E. AL=A(I)*Z(I)+A(I+1)*Z(I+1)
  007642  453277             STRALB   TMPPZ                         TMPPZ+ICR1:=AL,SCALING IS 2**15
  007643  370001             ENTBKB   1                             ICR1=ICR1+1 BUT THIS MEANS I=I+2
                *                         SINCE PSTP50 INCREMENTED  ICR1 ONCE ALREADY
  007644  347604             JP       PSTP50                        GO GET THE NEXT AMIN DESCRIPTOR
  007645  360001   PSTP54    ENTBK    1                             ICR1:=1
                *        THIS SECTION ADDS PBMIN*PYS + PAMIN*PZK
                *        AND PUTS THE RESULT IN THE PZKP1 VECTOR
  007646  133023   PSTP55    ENTALB   PZKP1-1                       AL:=(PZKP1+ICR1-1)
  007647  153276             ADDALB   TMPPZ-1                       AL:=AL+(TMPPZ+ICR1-1)
  007650  453023             STRALB   PZKP1-1                       PZKP1+ICR1-1:=AL
  007651  566713             ESK      PPR                           IF ICR1=PR+1 THEN SKIP ONE LINE
                *                         ELSE ICR1=ICR1+1 AND DO NEXT LINE
  007652  347646             JP       PSTP55                        CONTINUE
  007653  347654             JP       PSTP6                         GOTO NEXT STEP
                *
                *    -----------SECTION PSTP6----------
                *
                *          THIS IS THE 4TH MATRIX MULTIPLY IN THE CMPSTR
```

Example Data File for Using the ROPTSYS Computer Program.

```
DATA FOR OPTSYS FOR THE CHAPTER 2 NAVION EXAMPLE
L,IQ,INQ,IR,IOMF,ISS,IM,ITF1,DM,ITF3,IFDFW,IE,IDSTB,IDBG,ISET,ILNG,ISCLE,IMIN
1,1,0,0,0,0,0,0,0,0,3,0,0,0,2,0,1,1,1
NS,NC,NM,NPD,NO,NMOD,IREG
4,2,2,2,2,0,0
TS MATRIX(NS)
1.0,1.0,.01,.01
TC MATRIX(NC)
.01,1.0
TD MATRIX(NPD)
1.0,1.0
TM MATRIX(NM)
1.0,1.0,.01
TP MATRIX(NO)
1.0,1.0
F MATRIX(NS X NS)
-.045,.036,0,-32.2
-.37,-2.02,176.0,0
.00191,-.0396,-2.98,0
0,0,1.0,0
HO MATRIX(NO X NS)
1.0,0,0,0
0,-1.0,0,176.0
RL MATRIX(NO X NC)
0,0
0,0
(AX,AM) MATRIX(NO)
1,1
G MATRIX(NS X NC)
0,1.0
-.28,2.0
-11.0,0
0,0
B MATRIX(NC)
1,1
HM MATRIX(NM X NS)
1.0,0,0,0
0,-1.0,0,176.0
0,0,0,1.0
RN MATRIX(NM X NPD)
0,0
0,0
0,0
GAMMA MATRIX(NS X NPD)
.045,-.036
.37,2.02
-.00191,.0396
0,0
Q MATRIX(NPD)
24,6,9,98
R MATRIX(NM)
.318,.318,.00039
```

ROPTSYS User's Manual. (contd)

```
    FOR ONE REGULATOR SYNTHESIS USE
       F HO EM HS AM C B
    FOR FILTER SYNTHESIS USE
       F HO GAM Q R
    DSTAB R ENTER DIAGONAL ELEMENTS ONLY
    (AX AM) B Q ARE DIAGONAL OR FULL (SELECT (R)
    HO IS A VECTOR

 IF LAST CARD  COL 1 IS *  THEN CONTINUE TO NEXT CASE
 each case expects complete title lines  options  sizes  and matrices


SUBROUTINE LISTING
 SETUP USER CAN SET UP OPEN LOOP DYNAMICS MARTIX
 INNER OPTIMAL CONTROLLER/OPTIMAL ESTIMATOR DESIGN
 OMF OPTIMAL MODEL FOLLOWING DESIGN OPTION
 CDIV COMPLEX DIVISION
 RAPRINT MATRIX PRINTING
 RGAIN REFORMATS EIGENVALUES CALCULATES MODAL SUBMATRICES
 MULT MATRIX MULTIPLICATION
 MADD MATRIX ADDITION
 MTRN MATRIX TRANSPOSITION
 MINV MATRIX INVERSION
 SLOV SOLVES LYAPUNOV EQUATION FOR SS COVARIANCE
 MODE COMPUTES MODAL COORDINATE MATRICES
 NORM FORMATS EIGENVALUE/EIGENVECTOR OUTPUT
 TF COMPUTES TRANSFER FUNCTIONS
 CHECK CHECK CONSISTENCY OF INPUTS
 POLES PRINT OUT TF POLES
 ZEROS COMPUTE SISO TF ZEROS
 ACOMP COMPUTE ZEROS BY BROCKETT'S METHOD (3 SUBROUTINES)
 CCOMP
 SCL FUNCTION
 RESID COMPUTE (TIME RESPONSE) RESIDUES
 BALANC FIND EIGENVALUES AND EIGENVECTORS VIA
 QRTHES     THE QR ALGORITHM (6 SUBROUTINES)
 ORTRAN
 HQR2
 BALBAK
 HQR
 MODCOM CALCULATES THE MODAL FORM OF THE COMPENSATOR(HOLDRIDGE)
 MINCOM CALCULATES THE MINIMAL FORM OF THE COMPENSATOR(HOLDRIDGE)
```

ROPTSYS User's Manual.

```
C (VIII.83) print RIC all equation solution if IDEBUG  2
C  input free format  expect and neglect one line title at line 1 2 4
C  and before each matrix
C  output use E instead of D (C input compatible)
C
C   IMPLICIT REAL*8(A H O Z)
C
C
C
C
C          OPTSYS 6
C
C        JEFFERY A  QUIRK
C
C      VERSION   22 JULY 1981
C
C      REVISED BY RICK HOLDRIDGE 11 Mar 84
C
C
C
C
C  THIS VERSION OF OPTSYS WILL COMPUTE
C     OPTIMAL CONTROLLERS
C     OPTIMAL MODEL FOLLOWING CONTROLLERS
C     OPTIMAL ESTIMATORS
C     SYSTEM TRANSFER FUNCTIONS
C     SYSTEM EIGENVALUES  EIGENVECTORS  AND MODAL MATRICES
C  FEATURES
C     NON DIAGONAL A B Q MATRICES ARE ALLOWED
C  OPTIMAL CONTROLLER EQUATIONS
C     DYNAMICS  DX F*X+G*U
C     OUTPUT     Y HO*X+RL*U
C     WEIGHTING MATRICES  A AND B    J = INT(YT*A*Y+UT*B*U)
C     CONTROLLER GAINS  C               NOTE   U =  C*X
C  OMF CONTROLLER EQUATIONS
C     MODEL DYNAMICS  DXM FM*XM
C     DESIGN STATES    XD HS*X
C     WEIGHTING MATRICES  AM AND B
C     CONTROLLER GAINS  C
C  ESTIMATOR EQUATIONS
C     DYNAMICS  DX=F*X+GAM*W
C     MEASUREMENTS  Y HM*X+RN*W+V
C     WEIGHTING MATRICES  Q AND R
C     ESTIMATOR GAINS  K
C
C
C
C        USER'S MANUAL
C
C
C
C  INPUTS
C
C     line 1  title or comment for the data set
C     line 2  comment for the options
C     CARD 3 (free format)
C        IOL IQ INQ IR IOMF ISS IM ITF1 DUM ITF3 IE IDSTAB
C        IDEBUG ISET ILONG ISCALE IMIN
C
C        DEFAULT VALUES ARE ZERO
C         IOL=1 OPEN LOOP EIGENSYSTEM IS DESIRED
C            =2 OPEN LOOP EIGENSYSTEM ONLY (THEN TERMINATE)
C            =3 MODAL DISTRIBUTION MATRICES ONLY
C                (NO REGULATOR OR FILTER SYNTHESIS)
C         IQ=1 IF THE RMS VALUES OF THE CONTROL AND STATE ARE TO BE FOUND
C         INQ 0 IF A B Q AND R ARE DIAGONAL
C              1 IF ONLY B AND R ARE DIAGONAL
C              2 IF ONLY R IS DIAGONAL
C         IR 0 IF OPTIMAL FILTER AND REGULATOR EIGENSYSTEMS ARE TO BE FOUND
C             1 IF EXTERNAL C MATRIX IS SUPPLIED
C             2 IF EXTERNAL K IS SUPPLIED
C             3 IF EXTERNAL C AND K ARE SUPPLIED
C         IOMF  0 IF OPTIMAL CONTROL REGULATOR DESIGN IS DESIRED
C               1 IF OPTIMAL MODEL FOLLOWING REGULATOR DESIGN DESIRED
C         ISS 1 IF STEADY STATE VALUES (FOR A CONSTANT DISTURBANCE)
C                ARE TO BE DETERMINED
C         IM 1 IF MODAL CONTOL DISTRIBUTION AND GAINS ARE DESIRED
C         ITF1 0 IF NO OPEN LOOP TRANSFER FUNCTIONS ARE DESIRED
C               1 IF POLES  RESIDUES  AND ZEROS ARE DESIRED
C               2 IF POLES AND ZEROS ARE DESIRED
C               3 IF POLES AND RESIDUES ARE DESIRED
C         DUM 0
C         ITF3 0 IF NO COMPENSATOR TRANSFER FUNCTIONS ARE DESIRED
C                COMPENSATOR TRANSFER FUNCTIONS CAN BE COMPUTED ONLY
C                IF BOTH REGULATOR AND FILTER ARE SYNTHESIZED OR INPUT
C               1 IF POLES  RESIDUES  AND ZEROS ARE DESIRED
C               2 IF POLES AND ZEROS ARE DESIRED
C               3 IF POLES AND RESIDUES ARE DESIRED
C         IDFDW 0 UNKNOWN PARAMETER USED IN THE TF CALLS
C         IE 0 DEFAULT CRITERIA FOR ZERO VALUES IS 10E 06
C         IE-  SETS THE ZERO CRITERIA FOR THE MARKOV PARAMETERS OF THE
C              NUMERATOR POLYNOMIAL OF THE TRANSFER FUNCTIONS
C         IDSTAB 1 IF FILTER (OR REGULATOR) SYNTHESIS BY THE
C                  DESTABILIZATION TECHNIQUE IS DESIRED
C         IDEBUG 1 PRINTS THE EULER LAGRANGE EIGENSYSTEM PRIOR TO
C                  THE MODAL DECOMPOSITION PROCEDURE
C                2 PRINTS THE RICCATI EQUATION SOLUTION MATRIX
C         ISET 1 IF THE SETUP SUBROUTINE IS TO BE USED TO DETERMINE
C                THE F MATRIX
C         ILONG 0 DOES NOT PRINT OUT EIGENVECTORS MATRICES (WHEN IM 1)
C         ISCALE 0 NO SCALING IS DONE  MATRICES TS TC TD.TM.TP AREN'T ENTERED
C               =1 SCALING IS DONE  THE TS TC TD TM TP MATRICES MUST BE IN DATA
C
C         IMIN 0 THE MINIMAL FORM OF THE COMPENSATOR IS NOT FOUNE
C             =1 THE MINIMAL FORM OF THE COMPENSATOR IS CALCULATED
C
C     line 4  comment for the numbers
C     CARD 5 (free format)
C        NS NC NOB NG NOUT NMOD IREG
C
C         NS = NUMBER OF STATES (MAX 20)
C         NC = NUMBER OF CONTROLS (MAX 8)
C         NOB   NUMBER OF MEASUREMENTS (MAX 8)
C         NG = NUMBER OF DISTURBANCES (MAX 8)
C         NOUT = NUMBER OF OUTPUTS (MAX 8)
C         NMOD = NUMBER OF MODEL STATES (MAX 10)
C          IREG = 1 FOR REGULATOR SYNTHESIS ONLY
C               = 0 FOR BOTH REGULATOR AND FILTER SYNTHESIS
C
C     MATRIX CARDS
C        one line title
C        free format, by row, wrap around if needed, new line for next row
C
C        TS TC TD TM TP F DSTAB HO RL D C
C        FM (AX AM) G B GAM HM RN Q R K MC
C
C        DELETE THOSE MATRICES NOT REQUIRED
C        FOR OPTIMAL REGULATOR SYNTHESIS USE
C           F HO AX G B
```

Appendix J.

ROPTSYS Computer Program

This appendix includes the users manual for the ROPTSYS computer program, which is located on the FSD VAX at NASA Ames Research center, and an example data file. The data file was used to calculate the full order compensator in the Navion example in Chapter 2. The lengthy FORTRAN listing is not shown.

```
[illegible] ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION   3                    PAGE  165

 7968  033266  000000   PXEZCN   RESERV   RESERV'1D                    RESERVE MEMORY FOR THE FADER RAMPS
 7969  033267  000000   PYEZCN   RESERV   RESERV'1D
 7970  033270  000000   PZEZCN   RESERV   RESERV'1D
 7971  033271  000000   PXEOFF   RESERV   RESERV'1D
 7972  033272  000000   PYEOFF   RESERV   RESERV'1D
 7973  033273  000000   PZEOFF   RESERV   RESERV'1D
 7974  033274  000000   PXCLST   RESERV   RESERV'1D
 7975  033275  000000   PYCLST   RESERV   RESERV'1D
 7976  033276  000000   PZCLST   RESERV   RESERV'1D
 7977  033277  000000   TPPPZ    RESERV   RESERV'10D                   TEMPORARY USED IN PSTP52 AND PSTP53
 7978                 *
 7979                 *          CONSTANTS FOR CWSTW4 SUBROUTINE
 7980                 *
 7981  033311  000001   PONE     1D
 7982  033312  000002   PTWO     2D
 7983                 *
 7984                 *     COMPLEMENTARY FILTER FREQUENCIES AND DAMPINGS
 7985                 *
 7986  033313  000310   KFREQH   200D
 7987  033314  000620   KFREQX   400D                                   KALMAN FREQ 1000/RPS
 7988  033315  000620   KFREQY   400D
 7989  033316  000620   KFREQZ   400D
 7990  033317  001303   KZETAH   707D
 7991  033320  001303   KZETAX   707D                                   KALMAN DAMPING 1000/UNIT
 7992  033321  001303   KZETAY   707D
 7993  033322  001303   KZETAZ   707D
 7994  033323  037200   KERLIM   16000D                                 KALMAN CVELO LIMITER
 7995                 *
```

Memory Allocation. (contd)

```
IBISR ASSEMBLER, VERSION 1.0   NOVRAS 10/02/84   SECTION

033200  000100   PK64     DATA     64C                                    64 CYCLE FADE VALUE
033201  000200   PK128    DATA     128C                                   128 CYCLE FADE VALUE
                 $ THE FOLLOWING 22 VARIABLE DECLARATIONS MUST STAY IN
                 $ THE ORDER HERE TO MAKE THE MEASUREMENT SELECTION LOGIC
                 $ IN CPPSTR4 CORRECT
                 $ -----------------------------------------------------------
033202  000000   PDTHET   RESERV   RESERV'1D                              PERTURBATION PITCH ANGLE
033203  000000   PPHI     RESERV   RESERV'1D                              ROLL ANGLE
033204  000000   PPSI     RESERV   RESERV'1D                              YAW ANGLE
033205  000000   PPITRT   RESERV   RESERV'1D                              PITCH RATE
033206  000000   PROLRT   RESERV   RESERV'1D                              ROLL RATE
033207  000000   PYAWRT   RESERV   RESERV'1D                              YAW RATE
033210  000000   PAX      RESERV   RESERV'1D                              LONG. ACCELERATION
033211  000000   PAY      RESERV   RESERV'1D                              LATERAL ACCELERATION
033212  000000   PAZ      RESERV   RESERV'1D                              NORMAL ACCELERATION
033213  000000   PDVEL    RESERV   RESERV'1D                              PERTURBATION AIRSPEED
033214  000000   PVVEL    RESERV   RESERV'1D                              VERTICAL VELOCITY
033215  000000   PBALT    RESERV   RESERV'1D                              BAROMETRIC ALTITUDE
033216  000000   PXBOD    RESERV   RESERV'1D                              X POSITION IN HEADING INERTIAL FRAME
033217  000000   PYBOD    RESERV   RESERV'1D                              Y POSITION IN HEADING INERTIAL FRAME
033220  000000   PZBOD    RESERV   RESERV'1D                              Z POSITION IN HEADING INERTIAL FRAME
033221  000000   PXBDCT   RESERV   RESERV'1D                              INERTIAL X VEL. IN HEADING INERTIAL FRAME
033222  000000   PYBDCT   RESERV   RESERV'1D                              INERTIAL Y VEL. IN HEADING INERTIAL FRAME
033223  000000   PZBDCT   RESERV   RESERV'1D                              INERTIAL Z VEL. IN HEADING INERTIAL FRAME
033224  000000   PX       RESERV   RESERV'1D                              RUNWAY FRAME X POSITION
033225  000000   PY       RESERV   RESERV'1D                              RUNWAY FRAME Y POSITION
033226  000000   PXDOT    RESERV   RESERV'1D                              RUNWAY FRAME X INERTIAL VELOCITY
033227  000000   PYDOT    RESERV   RESERV'1D                              RUNWAY FRAME Y INERTIAL VELOCITY
                 $ -----------------------------------------------------------
033230  000000   PXREF    RESERV   RESERV'1D                              REFERENCE POSITION
033231  000000   PXBREF   RESERV   RESERV'1D                              ROTATED REFERENCE POSITION
033232  000000   PDELX    RESERV   RESERV'1D                              X POSITION ERROR
033233  000000   PXDP     RESERV   RESERV'1D                              X PROPORTIONAL CONTROL
033234  000000   PXDD     RESERV   RESERV'1D                              X DERIVATIVE CONTROL
033235  000000   PXCTH    RESERV   RESERV'1D                              X 2ND DERIVATIVE CONTROL(THETA FEEDBACK)
033236  000000   PXCPR    RESERV   RESERV'1D                              X 3RD DERIVATIVE CONTROL(PITCH RATE FEEDBACK)
033237  000000   PXDPID   RESERV   RESERV'1D                              X TOTAL PID CONTROL
033240  000000   PXCOLD   RESERV   RESERV'1D                              OLD X TOTAL PID CONTROL
033241  000000   PYREF    RESERV   RESERV'1D                              REFERENCE POSITION
033242  000000   PYBREF   RESERV   RESERV'1D                              ROTATED REFERENCE POSITION
033243  000000   PDELY    RESERV   RESERV'1D                              Y POSITION ERROR
033244  000000   PYDP     RESERV   RESERV'1D                              Y PROPORTIONAL CONTROL
033245  000000   PYDD     RESERV   RESERV'1D                              Y DERIVATIVE CONTROL
033246  000000   PYCPH    RESERV   RESERV'1D                              Y 2ND DERIVATIVE CONTROL(PHI FEEDBACK)
033247  000000   PYCRR    RESERV   RESERV'1D                              Y 3RD DERIVATIVE CONTROL(ROLL RATE FEEDBACK)
033250  000000   PYDPID   RESERV   RESERV'1D                              Y TOTAL PID CONTROL
033251  000000   PYDOLD   RESERV   RESERV'1D                              OLD Y TOTAL PID CONTROL
033252  000000   PZREF    RESERV   RESERV'1D                              REFERENCE POSITION
033253  000000   PDELZ    RESERV   RESERV'1D                              Z POSITION ERROR
033254  000000   PZDP     RESERV   RESERV'1D                              Z PROPORTIONAL CONTROL
033255  000000   PZCD     RESERV   RESERV'1D                              Z DERIVATIVE CONTROL
033256  000000   PZDPID   RESERV   RESERV'1D                              Z TOTAL PID CONTROL
033257  000000   PZCOLD   RESERV   RESERV'1D                              OLD Z TOTAL PID CONTROL
                 $                 COLSET
033260  000000   PXDI     RESERV   RESERV'2                               X INTEGRAL CONTROL
033262  000000   PYDI     RESERV   RESERV'2                               Y INTEGRAL CONTROL
033264  000000   PZDI     RESERV   RESERV'2                               Z INTEGRAL CONTROL
```

```
1819A ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION  1                    PAGE  163

 7876  033113  000325            DATA    213D,-76D,755D,2048D,229D
       033114  777663
       033115  001363
       033116  004000
       033117  000345
 7877  033120  000000            DATA    0,2048D,1024D,73D,186D
       033121  004000
       033122  003130
       033123  000111
       033124  000272
 7878  033125  000020  PYSSEL    DATA    16D,17D,18D,5D,4D,6D,1D,2D
       033126  000021
       033127  000022
       033130  000003
       033131  000004
       033132  000006
       033133  000001
       033134  000002
 7879  033135  000005  PR        5D
 7880  033136  000003  PRC       3D
 7881  033137  000010  PRR       8D
 7882  033140  000002  PACESC    DATA    2,1,1,1
       033141  000001
       033142  000001
       033143  000001
 7883  033144  000000  PRP1      0D
 7884  033145  106332  PLNGSC    36058D                LONG. CYCLIC OUTPUT SCALING TERM(SCALED 2**18)
 7885  033146  147075  PCOLSC    52797D                COLL. OUTPUT SCALING TERM(SCALED 2**18)
 7886  033147  155124  PLATSC    55892D                LAT. CYCLIC OUTPUT SCALING TERM(SCALED 2**18)
 7887  033150  214356  PYAWSC    71918D                YAW OUTPUT SCALING TERM(SCALED 2**18)
 7888  033151  000000  PTRMTH    0                     THETA AT ENGAGE
 7889  033152  000000  PTRMPH    0                     PHI AT ENGAGE
 7890  033153  000000  PTRMAS    0                     AIRSPEED AT ENGAGE
 7891  033154  000000  PTRMHD    0                     HDOT AT ENGAGE
 7892  033155  777667  PKXP      DATA    -20CD         X PROPORTIONAL GAIN (1 FPS PER FT ERROR = 1024)
 7893  033156  776027  PKYP      DATA    -1060D        Y PROPORTIONAL GAIN (1 FPS PER FT ERROR = 1024)
 7894  033157  774057  PKZP      DATA    -2000D        Z PROPORTIONAL GAIN (1 FPS PER FT ERROR = 1024)
 7895  033160  777753  PKXI      DATA    -20D          X INTEGRAL GAIN
 7896  033161  777667  PKYI      DATA    -20CD         Y INTEGRAL GAIN
 7897  033162  777715  PKZI      DATA    -50D          Z INTEGRAL GAIN
 7898  033163  774057  PKXD      DATA    -2000D        X DERIVATIVE GAIN (1 FPS PER FPS *1024)
 7899  033164  776027  PKYD      DATA    -1000D        Y DERIVATIVE GAIN (1 FPS PER FPS *1024)
 7900  033165  767153  PKZD      DATA    -4500D        Z DERIVATIVE GAIN (1 FPS PER FPS *1024)
 7901  033166  001750  PKXTH     DATA    1000D         X THETA GAIN(1 FPS PER DEG = 1048)
 7902  033167  777013  PKYPH     DATA    -500D         Y PHI GAIN(1 FPS PER DEG = 1048)
 7903  033170  005670  PKXPR     DATA    3000D         X PITCH RATE GAIN(1 FPS PER DEG/SEC = 1048)
 7904  033171  777013  PKYRR     DATA    -500D         Y ROLL RATE GAIN(1 FPS PER DEG/SEC *1048)
 7905  033172  000372  PXDTNT    DATA    250D          LONGITUDINAL STICK DETENT(1024/IN)
 7906  033173  000372  PYDTNT    DATA    250D          LATERAL STICK DETENT(1024/IN)
 7907  033174  000372  PZDTNT    DATA    250D          COLLECTIVE DETENT(1024/IN)
 7908  033175  000010  PR8       DATA    8D            8 CYCLE FADE VALUE
 7909  033176  000360  PR240     DATA    240D          15FT X POSITION INTEGRATOR HOLD THRESHOLD
 7910  033177  000040  PR32      DATA    32D           32 CYCLE FADE VALUE
```

```
1819A ASSEMBLER, VERSION 1.0   MOVF43  10/02/84   SECTION   3                    PAGE  162

          033021  000000
          033022  000000
          033023  000000
 7868     033024  000000   PZKP1    DATA     0,0,0,0,0
          033025  000000
          033026  000000
          033027  000000
          033030  000000
 7869     033031  715722   PAMIN    DATA     -25645D,57115D,29685D,31459D,23903D
          033032  157433
          033033  071765
          033034  079343
          033035  056537
 7870     033036  751556   PBMIN    DATA     -11409D,4027D,-12415D,-217D,933D,342D,1960D,-35796D
          033037  007673
          033040  767600
          033041  777446
          033042  001685
          033043  000526
          033044  003650
          033045  672053
 7871     033046  747060   DATA     -12751D,-13925D,-14764D,-235D,932D,150D,1513D,-32809D
          033047  744632
          033050  743123
          033051  777424
          033052  001844
          033053  000226
          033054  002751
          033055  677726
 7872     033056  108077   DATA     35903D,-8724D,16093D,+13D,-2021D,-1111D,-90D,6944D
          033057  756753
          033060  033415
          033061  777762
          033062  774032
          033063  775650
          033064  777845
          033065  015440
 7873     033066  700212   DATA     -8053D,168D,26173D,1D,131D,24D,52D,144D
          033067  000250
          033070  003075
          033071  000001
          033072  000203
          033073  000030
          033074  000084
          033075  000220
 7874     033076  813280   DATA     -59727D,19294D,1007D,109D,-12582D,2449D,-51995D,-30852D
          033077  045536
          033100  001757
          033101  000155
          033102  747331
          033103  004621
          033104  032344
          033105  703573
 7875     033106  770035   PICMIN   DATA     -3682D,3740D,2048D,-337D,2048D
          033107  007234
          033110  004000
          033111  777256
          033112  004000
```

```
1819A ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION   3                                   PAGE  161

 7820  032710  000000   OZCPID   RESERV   RESERV'1D                        Z TOTAL PID CONTROL
 7821  032711  000000   OZCCLD   RESERV   RESERV'1D                        CLD Z TOTAL PID CONTROL
 7822                 $           CBLSET
 7823  032712  000000   OXCI     RESERV   RESERV'2                         X INTEGRAL CONTROL
 7824  032714  000000   OYCI     RESERV   RESERV'2                         Y INTEGRAL CONTROL
 7825  032716  000000   OZCI     RESERV   RESERV'2                         Z INTEGRAL CONTROL
 7826  032720  000000   CXEZCN   RESERV   RESERV'1D                        RESERVE MEMORY FOR THE FADER RAMPS
 7827  032721  000000   OYEZCN   RESERV   RESERV'1D
 7828  032722  000000   OZEZCN   RESERV   RESERV'1D
 7829  032723  000000   OXECPF   RESERV   RESERV'1D
 7830  032724  000000   OYECPF   RESERV   RESERV'1D
 7831  032725  000000   OZECPF   RESERV   RESERV'1D
 7832  032726  000000   CXCLST   RESERV   RESERV'1D
 7833  032727  000000   CYCLST   RESERV   RESERV'1D
 7834  032730  000000   CZCLST   RESERV   RESERV'1D
 7835  032731  000000   TMPOZ    RESERV   RESERV'10D                       TEMPORARY USED IN CSTP52 AND CSTP53
 7836                 $
 7837                 $          CONSTANTS FOR CMPSTR SUBROUTINE
 7838                 $
 7839  032743  000001   ONE      1D
 7840  032744  000002   01%0     2D
 7841                 $ **********************************************************************
 7842                 $
 7843                 $          THIS CODE RESERVES THE CONSTANTS AND
 7844                 $          VARIABLES NEEDED BY THE CMSTR4 SUBROUTINE
 7845                 $          WHICH IS ASSEMBLED IN MEMORY AREA #1.
 7846                 $
 7847  032745  000000   PUC      RESERV   RESERV'4D                        SET POINT BIAS COMMAND TO ACTUATOR
 7848  032751  000000   PUR      RESERV   RESERV'4D                        REGULATOR COMMANDS TO ACTUATOR
 7849  032755  000000   PUT      RESERV   RESERV'4D                        TOTAL ACTUATOR COMMANDS
 7850                 $                                  PUT(1)=CELB(LONG CYCLIC)
 7851                 $                                  PUT(2)=CELC(COLLECTIVE)
 7852                 $                                  PUT(3)=CELS(LAT CYCLIC)
 7853                 $                                  PUT(4)=CELR(DIRECTIONAL)
 7854  032761  000000   PPLT     RESERV   RESERV'4D                        PILOT COMMANDS
 7855                 $                                  PPLT(1)=FOR-AFT STICK
 7856                 $                                  PPLT(2)=COLLECTIVE
 7857                 $                                  PPLT(3)=LEFT-RIGHT STICK
 7858                 $                                  PPLT(4)=RUDDER PEDALS
 7859  032765  000000   PYS      RESERV   RESERV'10D                       MEASUREMENTS
 7860  032777  000000   PYCES    RESERV   RESERV'4D                        DESIRED OUTPUT VECTOR
 7861  033003  661677   PYSCLE   DATA     -40000D,-200000D,400000D
                                                                          LNG. CYCLIC SCALING(20000=10FPS/IN)
       033004  730737
       033005  116100
 7862                 $                                        COLLECTIVE SCALING
 7863                 $                                        LATERAL CYCLIC SCALING
 7864  033006  747214   PNK      DATA     -12659D,-23659D,-823D
       033007  772772
       033010  776020
 7865  033011  003454            DATA     1836D,-20665D,550D
       033012  727906
       033013  001046
 7866  033014  771767            DATA     -3080D,-8970D,19430D
       033015  776176
       033016  036106
 7867  033017  000000   PZK      DATA     0,0,0,0,0
       033020  000000
```

Memory Allocation. (contd)

```
18198 ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION   3                                       PAGE 157

     032344  000000   NIRMIH    0                                        THETA AT ENGAGE
     032345  000000   NIRMAS    0                                        AIRSPEED AT ENGAGE
     032346  000000   NIRMHC    0                                        HDOT AT ENGAGE
                    $ THE FOLLOWING 12 VARIABLE DECLARATIONS MUST STAY IN
                    $ THE ORDER HERE TO MAKE THE MEASUREMENT SELECTION LOGIC
                    $ IN CMPSTR2 CORRECT
                    $ ----------------------------------------------------------
     032347  000000   NDTHET    RESERV   RESERV'1D                       PERTURBATION PITCH ANGLE
     032350  000000   NPHI      RESERV   RESERV'1D                       ROLL ANGLE
     032351  000000   NPSI      RESERV   RESERV'1D                       YAW ANGLE
     032352  000000   NPITRT    RESERV   RESERV'1D                       PITCH RATE
     032353  000000   NROLRT    RESERV   RESERV'1D                       ROLL RATE
     032354  000000   NYAWRT    RESERV   RESERV'1D                       YAW RATE
     032355  000000   NAX       RESERV   RESERV'1D                       LONG. ACCELERATION
     032356  000000   NAY       RESERV   RESERV'1D                       LATERAL ACCELERATION
     032357  000000   NAZ       RESERV   RESERV'1D                       NORMAL ACCELERATION
     032360  000000   NCVEL     RESERV   RESERV'1D                       PERTURBATION AIRSPEED
     032361  000000   NVVEL     RESERV   RESERV'1D                       VERTICAL VELOCITY
     032362  000000   NBALT     RESERV   RESERV'1D                       BAROMETRIC ALTITUDE
                    $ ==========================================================
     032363  000000   TMPN2     RESERV   RESERV'10D                      TEMPORARY USED IN NSTP52 AND NSTP53
                    $
                    $         CONSTANTS FOR CMSTR2 SUBROUTINE
                    $
     032375  000001   NONE      1D
     032376  000002   NTWO      2D
                    $
                    $ **************************************************************
                    $
                    $         THIS CODE RESERVES THE CONSTANTS AND
                    $         VARIABLES NEEDED BY THE CMPSTR3 SUBROUTINE
                    $         WHICH IS ASSEMBLED IN MEMORY AREA VI.
                    $
     032377  000000   OUC       RESERV   RESERV'4D                       SET POINT BIAS COMMAND TO ACTUATOR
     032403  000000   OUR       RESERV   RESERV'4D                       REGULATOR COMMANDS TO ACTUATOR
     032407  000000   OUT       RESERV   RESERV'4D                       TOTAL ACTUATOR COMMANDS
                    $                         OUT(1)=CELB(LONG CYCLIC)
                    $                         OUT(2)=CELC(COLLECTIVE)
                    $                         OUT(3)=CELS(LAT CYCLIC)
                    $                         OUT(4)=CELR(DIRECTIONAL)
     032413  000000   OPLT      RESERV   RESERV'4D                       PILOT COMMANDS
                    $                         OPLT(1)=FOR-AFT STICK
                    $                         OPLT(2)=COLLECTIVE
                    $                         OPLT(3)=LEFT-RIGHT STICK
                    $                         OPLT(4)=RUDDER PEDALS
     032417  000000   OYS       RESERV   RESERV'10D                      MEASUREMENTS
     032431  000000   OYDES     RESERV   RESERV'4D                       DESIRED OUTPUT VECTOR
     032435  730737   OYSCLE    DATA     -20000D,-20000D,20000D
                                                                         LNG. CYCLIC SCALING(20000=10FPS/IN)
     032436  730737
     032437  047040
                    $                                   COLLECTIVE SCALING
                    $                                   LATERAL CYCLIC SCALING
     032440  733416   ONN       DATA     -10673D,3507D,-1591D
     032441  006663
     032442  774710
     032443  003457             DATA     1839D,-20664D,549D
     032444  727507
```

Memory Allocation. (contd)

```
1819A ASSEMBLER, VERSION 1.0   MCVR43  10/02/84   SECTION   3                                          PAGE  156

 7644                        8                              NPLT(4)=RUDDER PECALS
 7645  032235  000000   NYS      RESERV   RESERV'10D                        MEASUREMENTS
 7646  032247  000000   NYCES    RESERV   RESERV'4D                         CESIRED CUTPUT VECTOR
 7647  032253  000000   NYACT    RESERV   RESERV'4C                         ACUAL OUTPUTS CORRESPONDING TO NYCES
 7648  032257  742547   NYSCLE   CATA     -15C00D,20000D                    COMMANC SCALING TERMS(20000=1 FPS PER INCH)
       032260  047040
 7649  032261  775614   NKINT    CATA     -1139D,515D                       INTEGRAL CONTROL GAINS
       032262  001003
 7650  032263  774474            CATA     -1731D,2576D
       032264  005020
 7651  032265  413627   NNK      CATA     -125032D,-32537D
       032266  700346
 7652  032267  714511            CATA     -26294D,91551D
       032270  262637
 7653  032271  000000   NZK      CATA     0,0,0
       032272  000000
       032273  000000
 7654  032274  000000   NZKP1    CATA     0,0,0
       032275  000000
       032276  000000
 7655  032277  744311   NAMIN    CATA     -14134D,34775D,28947D
       032300  103727
       032301  070423
 7656  032302  776215   NBMIN    CATA     -882D,-89943D,713C,26005D
       032303  520250
       032304  001311
       032305  062625
 7657  032306  767456            CATA     -43C5D,-61197D,-10863D,45634C
       032307  610362
       032310  752620
       032311  131102
 7658  032312  000631            CATA     4C9C,1320D,27053D,53016D
       032313  002450
       032314  064655
       032315  147430
 7659  032316  012545   NICMIN   CATA     5477C,10596D,16384D
       032317  024544
       032320  040000
 7660  032321  000000            CATA     0,16384D,-6644D
       032322  040000
       032323  763013
 7661  032324  000004   NYSSEL   CATA     4C,1D,11D,10D
       032325  000001
       032326  000013
       032327  000012
 7662  032330  000012   NYDSEL   CATA     1CC,11D                           CESIREC CCMMANDS ARE VELOCITY AND HCOT
       032331  000013
 7663  032332  000003   NRCATA   3D
 7664  032333  000002   NNC      2D
 7665  032334  000004   NNN      4D
 7666  032335  000002   NACESC   CATA     2,1
       032336  000001
 7667  032337  000004   NRP1     4D
 7668  032340  100332   NLNGSC   38030D                                     LONG.CYCLIC SCALING TERM(SCALED 2**18)
 7669  032341  147075   NCCLSC   52797D                                     COLL. SCALING TERM(SCALED 2**18)
 7670  032342  105124   NLATSC   35892D                                     LAT SCALING TERM(SCALED 2**18)
 7671  032343  214356   NYAWSC   71918D                                     YAW SCALING TERM(SCALED 2**18)
```

Memory Allocation.

```
10198 ASSEMBLER, VERSION 1.0   MOVP43  10/02/84   SECTION   3                                    PAGE  158

032142  000012  MYCSEL   DATA    10D,11D                         DESIRED COMMANDS ARE VELOCITY AND HDOT
032143  000013
032144  000002  MR       2D
032145  000002  MRC      2D
032146  000004  MRH      4D
032147  000001  MACESC   DATA    1,1
032150  000001
032151  000003  MRPI     3D
032152  106332  MLNGSC   36058D                                  LONG. CYCLIC OUTPUT SCALING TERM(SCALED 2**18)
032153  147075  MCOLSC   52797D                                  COLL. OUTPUT SCALING TERM(SCALED 2**18)
032154  155124  MLATSC   55892D                                  LAT. CYCLIC OUTPUT SCALING TERM(SCALED 2**18)
032155  214356  MYAWSC   71918D                                  YAW OUTPUT SCALING TERM(SCALED 2**18)
032156  000000  MIRMTH   0                                       THETA AT ENGAGE
032157  000000  MIRMAS   0                                       AIRSPEED AT ENGAGE
032160  000000  MIRMHC   0                                       HDOT AT ENGAGE
                *  THE FOLLOWING 12 VARIABLE DECLARATIONS MUST STAY IN
                *  THE ORDER HERE TO MAKE THE MEASUREMENT SELECTION LOGIC
                *  IN CMPSTR1 CORRECT
                *  ----------------------------------------------------------
032161  000000  MOTHET   RESERV  RESERV'1D                       PERTURBATION PITCH ANGLE
032162  000000  MPHI     RESERV  RESERV'1D                       ROLL ANGLE
032163  000000  MPSI     RESERV  RESERV'1D                       YAW ANGLE
032164  000000  MPITRT   RESERV  RESERV'1D                       PITCH RATE
032165  000000  MROLRT   RESERV  RESERV'1D                       ROLL RATE
032166  000000  MYAWRT   RESERV  RESERV'1D                       YAW RATE
032167  000000  MAX      RESERV  RESERV'1D                       LONG. ACCELERATION
032170  000000  MAY      RESERV  RESERV'1D                       LATERAL ACCELERATION
032171  000000  MAZ      RESERV  RESERV'1D                       NORMAL ACCELERATION
032172  000000  MDVEL    RESERV  RESERV'1D                       PERTURBATION AIRSPEED
032173  000000  MVVEL    RESERV  RESERV'1D                       VERTICAL VELOCITY
032174  000000  MBALT    RESERV  RESERV'1D                       BAROMETRIC ALTITUDE
                *  ----------------------------------------------------------
032175  000000  TMPM2    RESERV  RESERV'10D                      TEMPORARY USED IN MSTP52 AND MSTP53
                *
                *           CONSTANTS FOR CMPSTR SUBROUTINE
                *
032207  000001  MONE     1D
032210  000002  MTWO     2D
                *  ***************************************************************
                *
                *           THIS CODE RESERVES THE CONSTANTS AND
                *           VARIABLES NEEDED BY THE CMSTR2 SUBROUTINE
                *           WHICH IS ASSEMBLED IN MEMORY AREA 01.
                *
032211  000000  MUI      RESERV  RESERV'4D                       INTEGRAL CONTROL COMPONENTS
032215  000000  MUC      RESERV  RESERV'4D                       SET POINT BIAS COMMAND TO ACTUATOR
032221  000000  MUR      RESERV  RESERV'4D                       REGULATOR COMMANDS TO ACTUATOR
032225  000000  MUT      RESERV  RESERV'4D                       TOTAL ACTUATOR COMMANDS
                *                          MUT(1)=DELB(LONG CYCLIC)
                *                          MUT(2)=DELC(COLLECTIVE)
                *                          MUT(3)=DELS(LAT CYCLIC)
                *                          MUT(4)=DELR(DIRECTIONAL)
032231  000000  MPLT     RESERV  RESERV'4D                       PILOT COMMANDS
                *                          MPLT(1)=FOR-AFT STICK
                *                          MPLT(2)=COLLECTIVE
                *                          MPLT(3)=LEFT-RIGHT STICK
```

Hover Controller Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   HCVR43  10/02/84   SECTION   1                                   PAGE  100

007733  566716            ESK      PACTRL                        IF PSK=PACTRL SKIP THE NEXT LINE
                  *                  ELSE ICR1=ICR1+1 AND DO NEXT LINE
007734  347726            JP       PSTF70                        CONTINUE
007735  347730            JP       PSTF8                         WRITE THE DATA OUT
                  *
                  *  ------------SECTION PSTP8------------
                  *
                  *          HERE GOES THE SECTION(PSTP8) WHICH WRITES PUT(ACTUATOR
                  *          COMMANDS) TO THE LOCATION  WHICH SENDS PUT TO THE
                  *          ACTUATORS.
                  *                          PSTP8
007736  122755  PSTP8     ENTAL    PUT                           AL:=PUT(1),LONG CYCLIC
007737  243145            MULAL    PLNGSC                        SCALE FOR OUTPUT(1IN/140,85BITS)
007740  400072            STRAU    OBF1                          STORE PUT(1) TO ECS PITCH COMMAND
007741  122756            ENTAL    PUT+1                         AL:=PUT(2),COLLECTIVE
007742  243146            MULAL    PCOLSC                        SCALE FOR OUTPUT(1IN/206,24BITS)
007743  460075            STRAU    OBF4                          STORE PUT(2) TO ECS COLLECTIVE COMMAND
007744  122757            ENTAL    PUT+2                         AL:=PUT(3), LATERAL CYCLIC
007745  243147            MULAL    PLATSC                        SCALE FOR OUTPUT(1IN/218,33BITS)
007746  400073            STRAU    OBF2                          STORE PUT(3) TO ECS ROLL COMMAND
                  *          ENTAL'PUT+3 AL:=PUT(4), DIRECTIONAL COMMAND
                  *          MULAL'PYAWSC      SCALE FOR OUTPUT(1IN/280,93BITS)
                  *          STRAU'OBF3  STORE PUT(4) TO ECS YAW COMMAND
                  *
                  *  ------------SECTION PSTP9
                  *
                  *          THIS SECTION OF CMPSTR SETS THE PRESENT COMPENSATOR
                  *          STATES(PZKP1) TO OLD STATES(PZK)
                  *
007747  307201  PSTP9     ENTICR   1                             USE ICR1 AS INDEX
007750  360001            ENIBK    1                             ICR1=0  (I=0)
007751  133023  PSTP90    ENTALB   PZKP1-1                       AL:=(PZKP1+ICR1)
007752  453016            STRALB   PZK-1                         PZK+ICR1-1:=AL
007753  566713            ESK      PRR                           IF ICR1=PRR THEN SKIP 1 LINE
                  *                  ELSE ICR1=ICR1+1  AND DO NEXT LINE
007754  347751            JP       PSTP90                        CONTINUE
007755  556720            IJP      CMPSTR4                       END THE CONTROLLER SUBROUTINE NUMBER 4
```

Hover Controller Subroutine. (contd)

```
1819A ASSEMBLER, VERSION 1.0   MOVR43  10/02/84   SECTION   1                                   PAGE   55

                          *            SUBROUTINE, I.F.
                          *                  PUR = ICFIN * PZKP1
        007654  507310   PSTP6   ENTSR   10                                USE MEMORY BANK 0
        007655  507201           ENTICR  1                                 USE ICR1 AS THE INDEX REG.(B)
        007656  360000           ENTBK   0                                 INITIALIZE THE B REG. TO 0
        007657  400002           STEZ    2                                 ICR2=0, THE COLUMN INDEX
        007660  400003           STEZ    3                                 ICR3=0, THE ROW INDEX
        007661  507313           ENTSR   13                                USE MEMORY BANK 3
        007662  123135           ENTAL   PR                                AL:=NUM OF COMPENSATOR STATES
        007663  717776           ADDALK  -1                                AL:=PR-1
        007664  507310           ENTSR   10                                USE MEMORY BANK 0 SINCE ORWEND IS THERE
        007665  440711           STRAL   PRWEND                            PRWEND:=PR-1
        007666  507313           ENTSR   13                                BACK TO MEMORY BANK 3 FOR DATA
        007667  700000   PRXRW3  ENTALK  0                                 AL:=0
        007670  507400           XFRLU                                     AU:=AL=0
        007671  500600   PRXCL3  STRA    TPP                               TPP:=A
        007672  031650
        007673  507202           ENTICR  2                                 GET THE COLUMN INDEX
        007674  370001           ENTBKB  1                                 INCREMENT COLUMN INDEX(ICR2=ICR2+1)
        007675  133023           ENTALB  PZKP1-1                           AL:=(PZKP1+ICR2-1)
        007676  507201           ENTICR  1                                 USE ICR1 AS INDEX REG.(ELEMENT INDEX)
        007677  253106           MULALB  PTCFIN                            A:=AL * (PTCFIN+ICR1)
        007700  201650           ADDA    TPP                               A:=A + (TPP,TMP+1)
        007701  505300           SKPNOV                                    CHECK FOR OVERFLOW
        007702  504000           NOOP                                      CALL OVERFLOW HANDLER
        007703  566711           ESK     PRWEND                            IF ICR1=PRWEND THEN SKIP NEXT LINE
                          *                        ELSE DO NEXT LINE AND ICR1=ICR1+1
        007704  347671           JP      PRXCL3                            JUMPS TO NEXT COLUMN OF PTCFIN
                          *                        AND NEXT ROW ELEMENT OF PZKP1
        007705  370001           ENTBKB  1                                 ICR1=ICR1+1
        007706  507203           ENTICR  3                                 USE ICR3 AS INDEX REG.(ROW INDEX)
        007707  504702           LRTA    2                                 SCALING FOR PSTP6 IS 2**16
        007710  472751           STRAUB  PUR                               PUR+ICR1:=(AU),SCALING IS 2**16
        007711  566715           ESK     PNCPI                             IF ICR3=PNCPI THEN SKIP NEXT LINE
                          *                        ELSE DO NEXT LINE AND ICR3=ICR3+1
        007712  347714           JP      PRSET3                            GO RESET THE COUNTER
        007713  347724           JP      PSTP7                             GO TO NEXT MATRIX MULT IF DONE
        007714  507202   PRSET3  ENTICR  2                                 SAVE ROW INDEX AND START USING COLUMN INDEX
        007715  360000           ENTBK   0                                 RESET COLUMN INDEX TO ZERO
        007716  507310           ENTSR   10                                USE BANK 0 SINCE PRWEND AND PNCTRL ARE THERE
        007717  126711           ENTAL   PRWEND                            AL:=PRWEND
        007720  160713           ADDAL   PRR                               AL:=PRWEND+PRR=PRWEND+#OF COLUMNS
        007721  446711           STRAL   PRWEND                            PRWEND:=PRWEND+PNCTRL
        007722  507313           ENTSR   13                                USE BANK 3 FOR DATA
        007723  347667           JP      PRXRW3                            DO THE NEXT ROW
                          *
                          *      -----------SECTION PSTP7-------------
                          *
                          *            THIS STEP DOES THE ADDITION PUT = PUC + PUR
                          *
        007724  507201   PSTP7   ENTICR  1                                 USE ICR1
        007725  360001           ENTBK   1                                 (ICR1(B0) MEANS (I=1)
        007726  132744   PSTP70  ENTALB  PUC-1                             AL:=(PUC+ICR1-1)
        007727  152750           ADDALB  PUR-1                             AL:=AL+(PUR+ICR1-1)
        007730  505300           SKPNOV                                    SKIP IF NO OVERFLOW
        007731  504000           NOOP                                      CALL THE OVERFLOW HANDLER
        007732  452754           STRALB  PUT-1                             PUT:=AL, PUT(I)=PUC(I)+PUR(I)
```

Appendix K.

RSANDY Computer Program

The RSANDY computer program is a modified version of the SANDY computer program written by Uy-Loi Ly as part of his PhD dissertation at Stanford [3] and later modified first by Gardner [16]. It is stored on the FSD VAX at NASA Ames Research Center. Ly also wrote a user's guide for the SANDY program.[8] This appendix gives the input format changes to make the SANDY user's guide correct for the RSANDY program. The major capabilities added by the RSANDY program are:

- an optional gradient step-size reducer from Gardner [16]
- a linear discrete model of the closed loop system can be created for later simulation studies
- a leading free line and free lines before all the data items are included to help in documenting the data files

The changes which follow apply to page numbers in the SANDY User's Guide. Other than these changes, the program is exactly like the SANDY program. Also included here is an example data set for running the program. This data was used to find the reduced order Navion compensator in Chapter 2. In the following changes, the new variables needed by the program are italicized.

Change 1, page 117a. Running the RSANDY Program

@RSANDY Infile Outfile Simfile

where:

RSANDY- A VMS command file which runs the RSANDY.EXE file with Infile, Outfile, and Simfile as data files.

Infile- A file containing the input data; an example is shown at the end of this appendix.

Outfile- A filename where the program will write the output.

Simfile- A filename written by program, if IPLOT=1, which contains the linear simulation models used by the SIMPLOT program, described in Appendix M.

Change 2, page 119a. Item 1

Np,n,m,m',p,p',p",r,flag,*NNS,IPPSS,ICF,ISS*

where:

NNS- Set to 0.

IPPSS- Set to 0.

ICF- Set to 0.

ISS- Set to 0.

Change 3, page 119a. Item 2

Maxfn,Nvar,Tol,*MSTEP*,Nlinear,Tf,Print,*IDPRN,ICLPRN,MAPRN,IPLOT, DT,IBG*

where:

MSTEP- Maximum step size for the gradient algorithm. Start at 100 and make it smaller if there are convergence problems.

IDPRN- = 0 for no input data printout.

ICLPRN- = 0 for no closed loop data printout.

MAPRN- = 0 for no modal analysis printout.

IPLOT- = 0 for not creating a simulation model. = N for creating a simulation model of the N^{th} plant condition.

DT- Cycle time for the discrete simulation model. Rule of Thumb, $DT \approx .2(\frac{2\pi}{\omega_{fastest}})$.

Change 4, page 126. Add Item 9

Data: *XO*

Description:

XO- A vector with $(n + r + m')$ zeros.

RSANDY Example Data File.

```
DATA FOR RSANDY FOR THE CHAP 2 EX. THE NAVION AT 100 KNOTS(REDUCED ORDER COMP)
@PLNT N,M,M' P,P' P'' R FLAG,NNS,IPPSS,ICE ISS
    1,4,2, 4,2, 2, 5,2, 0, 0, 0, 0,0,0
Maxfn,Nvar,Tol,MSTEP,NLINEAR,Tf,Princ IDPRN ICLPRN,MAPRN,IPLOT, DT IBG
   00,    8,1.0E-01,100,0.999 00,    1,    0,    0,    0, 05,1
PROBABILITY DENSITY OF PLANT CONDITION 1
1 0
F MATRIX(N X N)U W,Q,THETA
 045, 036,0,-32 2
-37 -2 02 176 0,0
 00191, -0396,-2 98,0
0 0,1 0,0
G MATRIX( n X m )
0 1 0
-28 2,0
-11 0,0
0,0
GAMMA MATRIX( n X m' )
 045, -036,0,0
 37 2,02,0,0
 -00191, 0396,0,0
0,0 0,0
HS MATRIX( p X n )VELOCITY,HDOT
1 0,0,0,0
0 1 0 0,176 0
DSU MATRIX( p X m )
0 0
0 0
DSW MATRIX( p X m' )
0 0 1 0,0
0 0 0,1 0
HC MATRIX( p' x n )VELOCITY,HDOT
1 0 0,0,0
0 -1 0 0,176 0
DCU MATRIX( p' x m )
0 0
0 0
DCW MATRIX( p' x m' )
0 0 0 0 0
0 0 0 0 0
HP MATRIX( p'' x n )U,W,Q,THETA,HDOT
1 0 0 0 0
0 1 0,0,0
0 0 57 3 0
0 0 0,57 3
0 -1 0 0,176 0
DPU MATRIX( p'' x m )
0 0
0 0
0 0
0 0
0 0
DPW MATRIX( p'' X m' )
0 0 0,0
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0,0
UW NOISE  TYPE ORDER RMS
2 1 2 3
FILTER CONSTANT
 43
WW NOISE  TYPE ORDER RMS
2 1 2 3
FILTER CONSTANT
1 06
U MEAS  NOISE  TYPE,ORDER,RMS
2 1,1 0
FILTER CONSTANT(1HZ)
6 28
HDOT MEAS  NOISE  TYPE,ORDER,RMS
2 1,1 0
FILTER CONSTANT
6 28
ALPHA
0 00
Q MATRIX( p' x p' )VELOCITY,HDOT
1 0 0
0 1 0
R MATRIX( m x m )ELEVATOR,THROTTLE
10000 0,0
0,1 0
 A MATRIX
0 0000E+00   1 000    -13 59      -16 93
 B MATRIX
-0 1366      0 2124
2 676       -3 241
 C MATRIX
0 0000E+00 -0 1000E-01
-13 14      -1 228
IDVAR( Nvar )
3 4 5,6,7,8,11,12
ITEM 9  XO
0 0,0,0,0,0,0,0,0,0,0 0 0 0,0
```

Appendix L.

SETPNT Computer Program

This appendix lists the computer program which calculates the feedforward matrix described in Appendix D. This version does not have the G_z matrix. The data formats and data sequence are described at the beginning of the program. Also included is an example data set (the full-order Navion of Chapter 2).

SETPNT Computer Program.

```
*****************************************************************

 THIS PROGRAM TAKES, AS INPUT, A DYNAMIC SYSTEM AND FINDS THE
 "FEEDFORWARD MATRIX" WHICH WHEN MULTIPLIED BY A DESIRED OUTPUT
 GIVES THE STEADY CONTROL FOR THE NEW EQUILIBRIUM POINT DEFINED
 BY THE DESIRED OUTPUT  YD   IT ALSO FINDS THE TRANSFER FUNCTIONS
 OF THE COMPENSATOR

 THE DYNAMIC SYSTEM IS DEFINED BY THE FOLLOWING SET OF EQUATIONS

            XDOT = F * X  +  G * U                PLANT
              YS = HS * X  +  DSU * U             SENSORS
            ZDOT = A * Z  +  B * YS               COMPENSATOR DYNAMICS
               U = C * Z  +  D * YS  +  UC        CONTROLS

 WHERE THE DESIRE IS TO FIND US AS A FUNCTION OF THE SYSTEM MATRICES
 AND THE DESIRED OUTPUT VECTOR  YD

             YD = HD * XT  +  LD * UC

           WHERE   XT= X
                       Z

 SO AT STEADY STATE      UC = RN* YD
                         XT(STEADY STATE) = RM * YD

           WHERE M AND N COME FROM THE INVERSE OF THE FOLLOWING
                 MATRIX
                                                          1
               | F+G*D*HS           G*C          G     |   :  0  :
               | B*(IP+DSU*D)*HS   A+B*DSU*C   B*DSU   | * :  0  :
   | RM |      |                                       |   :     :
   | RN |      |                   HD            LD    |   : IM  :
               |_                                      |   :     :

 THE FOLLOWING IS THE DATA REQUIRED BY THE PROGRAM   THE DATA MUST
 BE IN THE ORDER SHOWN   THE DATA IS ASSUMED TO BE IN FOR003 DAT

      DESCRIPTION OF N,M P R(ONE LINE)
      N M P R
            WHERE  N IS THE NUMBER OF STATES
                   M IS THE NUMBER OF CONTROLS AND THE NUMBER OF DESIRED
                                                                 OUTPUTS
                   P IS THE NUMBER OF MEASUREMENTS
                   R IS THE ORDER OF THE COMPENSATOR
      TITLE OF F MATRIX
      F MATRIX(N X N) ENTERED BY ROW
      TITLE OF G MATRIX
      G MATRIX(N X M) ENTERED BY ROW
      TITLE OF HS MATRIX
      HS MATRIX(P X N) ENTERED BY ROW
      TITLE OF DSU MATRIX
      DSU MATRIX(P X M) ENTERED BY ROW
      TITLE OF A MATRIX
      A MATRIX(R X R) ENTERED BY ROW
      TITLE OF B MATRIX
      B MATRIX(R X P) ENTERED BY ROW
      TITLE OF C MATRIX
      C MATRIX(M X R) ENTERED BY ROW
C         TITLE OF D MATRIX
C         D MATRIX(M X P) ENTERED BY ROW
C         TITLE OF HD MATRIX
C         HD MATRIX(M X N+R) ENTERED BY ROW
C         TITLE OF LD MATRIX
C         LD MATRIX(M X M) ENTERED BY ROW
C
C*****************************************************************
      IMPLICIT REAL*8(A H O Z)
      REAL*8 LD(20 20)
      INTEGER P R
      CHARACTER*20 FNAME
      DIMENSION F(20 20),G(20 20) HS(20 20) DSU(10 20),
     *A(20,20),B(20 20),C(20 20) D(20,20) HD(20,40)
     *BIGM(44 44) TMP(44 44) IPS(44) RM(40,20) RN(20 20)
     *BIGMIN(44,44),WKAREA(2068) ADIG(20 20),BDIG(20,10) CDIG(4 20)
     *BTMP(44 4) BTMP1(44 4) TMP1(20 20) TMP2(20,20) TMP3(20 20)
     *TMP4(20,20) TMP5(20,20) TMP6(20,20),TMP7(20,20),
     *TMP8(20 20),TMP9(20 20),TMP10(20,20),TMP11(20,20)
C
C  INPUT THE DATA
C
C  N M.P.R
 1020 FORMAT('O',2X 'N=',I2,'  M=',I2,'  P=',I2,'  R=',I2,
     *'  SAMPLE PERIOD(SEC)= ',F7 4)
 1000 FORMAT('O',// 2X,'THE BIGM MATRIX IS SINGULAR')
 1010 FORMAT('O' // 2X,'THE BIGM MATRIX IS ILL CONDITIONED')
 1030 FORMAT(1X,// 2X 'THE BIGM MATRIX IS')
      READ(3 *)
      READ(3,*)N.M.P.R.T
      WRITE(6 1020)N.M.P.R.T
      NPLR=N+R
      CALL DATAIN(N M P,R NPLR FNAME F,G HS,DSU A B C D HD LD)
C
C  THESE STATEMENTS ARE COMMENTED BECAUSE THE THEORY BEHIND
C  GETTING M AND N IS WRONG
C
C  FILL THE BIGM MATRIX USING THE DATA
C
      NSUM N+R+M
      CALL FILLEM(N M.P.R.NPLR.NSUM.F.G HS.DSU A.B.C.D HD
     *.LD BIGM.TMP1 TMP2 TMP3.TMP4 TMP5.TMP6 TMP7 TMP8.TMP9 TMP10
     *.TMP11)
      WRITE(6 1030)
      CALL PRNTM(BIGM NSUM NSUM NSUM)
C
C  INVERT THE BIGM MATRIX
C
      CALL LINV2F(BIGM NSUM NSUM BIGMIN,5,WKAREA IER)
      IF (IER EQ 129) WRITE(6,1000)
      IF (IER EQ 131) WRITE(6 1010)
C      CALL MATPNT(BIGMIN NSUM.NSUM,'INVERSE(BIGM)')
C
C  CHECK CORRECTNESS OF THE INVERSION
C
C      CALL MULT(BIGMIN BIGM TMP,NSUM NSUM NSUM)
C      CALL MATPNT(TMP NSUM.NSUM,'IDENTITY(NSUM)')
C
C  EXTRACT THE DATA AND PRINT THE RESULTS
C
      CALL GETMN(N M P R.NPLR NSUM.BIGMIN BTMP BTMP1 RM RN)
C
C  CALL THE SUBROUTINE WHICH DOES THE MATRIX POLYNOMIAL AND
C  PARTIAL FRACTION EXPANSION OF THE COMPENSATOR   THIS ALSO
```

SETPNT Computer Program. (contd)

```
FINDS THE ZEROS OF THE TRANSFER FUNCTIONS OF THE COMPENSATOR
   CALL TRANAL(A B C M P R)

THIS SUBROUTINE CALL FINDS THE DISCRETE FORM OF THE COMPENSATOR
AND ALSO FINDS THE MINIMAL FORM OF THE DISCRETE COMPENSATOR

   CALL DIGITL(A B C ADIG BDIG CDIG T R M P)
   STOP
   END

*****************DATAIN*************************************************

THIS SUBROUTINE READS THE DATA FROM A DATA FILE IN
FREE FORMAT   IT FIRST ASKS FOR THE NAME OF THE FILE

************************************************************************
      SUBROUTINE DATAIN(N M P R NPLR FNAME F G HS DSU A B
     *C D HD LD)
      IMPLICIT REAL*8(A H O Z)
      REAL*8 LD(M M)
      INTEGER P R
      CHARACTER*20 FNAME
      DIMENSION F(N N) G(N M) HS(P N) DSU(P M)
     *A(R R) B(R P) C(M R) D(M P) HD(M NPLR)
1000  FORMAT(' INPUT FILE NAME  '\) USED FOR IBM PC VERSION
1010  FORMAT(A)
1030  FORMAT('O' 2X 'HD(' I2 I2 ')  ' F11 5)

 PROMPT THE USER FOR THE FILENAME

      WRITE(6 1000)
      READ(6 1010)

 OPEN UNIT 3 FOR INPUT

      OPEN(3 FILE FNAME)

 READ THE DATA

      READ(3 *)
      DO 100 I=1 N
100      READ(3 *) (F(I J) J=1 N)
      CALL MATPNT(F N N 'F')
      READ(3 *)
      DO 110 I=1 N
110      READ(3 *) (G(I J) J 1 M)
      CALL MATPNT(G N M 'G')
      READ(3 *)
      DO 120 I=1 P
120      READ(3 *) (HS(I J) J 1 N)
      CALL MATPNT(HS P N 'HS')
      READ(3 *)
      DO 130 I=1 P
130      READ(3 *) (DSU(I J) J=1 M)
      CALL MATPNT(DSU P M 'DSU')
      READ(3 *)
      DO 140 I=1 R
140      READ(3 *) (A(I J) J 1 R)
      CALL MATPNT(A R R 'A')
      READ(3 *)
      DO 150 I=1 R
150      READ(3 *) (B(I J) J=1 P)
      CALL MATPNT(B R P 'B')
      READ(3 *)
      DO 160 I 1 M
160      READ(3 *) (C(I J) J 1 R)
      CALL MATPNT(C M R 'C')
      READ(3 *)
      DO 170 I 1 M
170      READ(3 *) (D(I J) J 1 P)
      CALL MATPNT(D M P 'D')
      READ(3 *)
      NPR N+R
      DO 180 I 1 M
180         READ(3 *) (HD(I J) J 1 NPR)
      CALL MATPNT(HD M NPR 'HD')
      READ(3 *)
      DO 190 I 1 M
190      READ(3 *) (LD(I J) J 1 M)
      CALL MATPNT(LD M M 'LD')
      RETURN
      END
C*************************FILLEM*****************************************
C
C  THIS SUBROUTINE USES THE DATA READ IN IN THE SUBROUTINE DATAIN
C  TO FILL THE BIGM MATRIX
C
C***********************************************************************
      SUBROUTINE FILLEM(N M P R NPLR NSUM F G HS DSU A B C D HD
     * LD BIGM TMP1 TMP2 TMP3 TMP4 TMP5 TMP6 TMP7 TMP8 TMP9 TMP10
     * TMP11)
      IMPLICIT REAL*8(A H O Z)
      REAL*8 LD(M M)
      INTEGER P R
      DIMENSION F(N N) G(N M) HS(P N) DSU(P M)
     *A(R R) B(R P) C(M R) D(M P) HD(M NPLR)
     *BIGM(NSUM NSUM) TMP1(M N) TMP2(N N)
     *TMP3(N N) TMP4(N R) TMP5(P P) TMP6(P N)
     *TMP7(R N) TMP8(P R) TMP9(R R) TMP10(R R)
     *TMP11(R M)
1000 FORMAT(10(2X 1PE10 3) / 2X 10(2X 1PE10 3))
1010 FORMAT('O' 2X 'THE BIGM MATRIX IN FILLEM IS')
C  TMP1 D*HS
      CALL MULT(D HS TMP1 M P N)
C  TMP2=G*D*HS-G*TMP1
      CALL MULT(G TMP1 TMP2 N M N)
C  TMP3 F+G*D*HS=F+TMP2
      CALL MADD(F TMP2 TMP3 N N 1 0)
C  LOAD THE BIGM MATRIX
      DO 100 I=1 N
      DO 100 J 1 N
100      BIGM(I J) TMP3(I J)
C  TMP4 G*C AND LOAD THE BIGM MATRIX
      CALL MULT(G C TMP4 N M R)
      DO 110 I=1 N
      DO 110 J 1 R
110      BIGM(I N+J)=TMP4(I J)
C  LOAD THE BIGM MATRIX
      NPR N+R
      DO 120 I 1 N
      DO 120 J 1 M
120      BIGM(I NPR+J) G(I J)
C  TMP5 DSU*D
      CALL MULT(DSU D TMP5 P M P)
C  TMP5 TMP5+I DSU*D+I
      DO 130 I 1 P
130      TMP5(I I) TMP5(I I)+1 0
```

SETPNT Computer Program. (contd)

```
      TMP6 TMP5*HS (I+DSU*D)*HS
         CALL MULT(TMP5 HS TMP6 P P N)
      TMP7 B*TMP6 B*(I+DSU*D)*HS
         CALL MULT(B TMP6 TMP7 R P N)
      LOAD THE BIGM MATRIX
         DO 140 I 1 R
         DO 140 J 1 N
140         BIGM(N+I J) TMP7(I J)
      TMP8 DSU*C
         CALL MULT(DSU C TMP8 P M R)
      TMP9 B*TMP8 B*DSU*C
         CALL MULT(B TMP8 TMP9 R P R)
      TMP10 A+TMP9=A+B*DSU*C
         CALL MADD(A TMP9 TMP10 R R 1 0)
      LOAD THE BIGM MATRIX
         DO 150 I 1 R
         DO 150 J 1 R
150         BIGM(N+I N+J) TMP10(I J)
      TMP11 B*DSU
         CALL MULT(B DSU TMP11 R P M)
      LOAD THE BIGM MATRIX
         DO 160 I=1 R
         DO 160 J=1 M
160         BIGM(N+I NPR+J) TMP11(I J)
         DO 170 I 1 M
         DO 170 J 1 NPR
170         BIGM(NPR+I J) -HD(I J)
         DO 180 I 1 M
         DO 180 J 1 M
180         BIGM(NPR+I NPR+J) LD(I J)

      PRINT THE BIGM MATRIX

          WRITE(6 1010)
          DO 190 I 1 NSUM
190          WRITE(6 1000) (BIGM(I J) J 1 NSUM)
         RETURN
         END
*************************MATPNT**********************************

      THIS SUBROUTINE PRINTS A MATRIX AND LABELS THE PRINTING OF
      THE MATRIX

                    F    THE MATRIX TO BE PRINTED(LIMITED TO 20X20)
                    N    THE ROW DIMENSION OF THE MATRIX
                    M    THE COLUMN DIMENSION OF THE MATRIX
                    NAME   THE ASCII NAME OF THE MATRIX

*****************************************************************
         SUBROUTINE MATPNT(F N M NAME)
         IMPLICIT REAL*8(A H O Z)
         CHARACTER*(*) NAME
         DIMENSION F(N,M)
1000  FORMAT('0',//,2X,'THE ',A,' MATRIX IS ')
1010  FORMAT(10(2X,1PE10 3) /,2X,10(2X,1PE10 3))
         WRITE(6,1000)NAME
         DO 100 I=1 N
100         WRITE(6,1010) (F(I J) J=1 M)
         RETURN
         END
C
C
         SUBROUTINE MULT(A B C N1 N2 N3)
C
C  THIS SUBROUTINE MULTIPLIES MATRIX A(N1 X N2) BY MATRIX
C  B(N2 X N3)
C                      C A*B
         IMPLICIT REAL*8 (A H O Z)
         DIMENSION A(N1 N2) B(N2 N3) C(N1 N3)
         DO 7901 I 1 N1
         DO 7901 J 1 N3
         TEMP 0
         DO 7900 K 1 N2
 7900 TEMP TEMP+A(I K)*B(K J)
 7901 C(I J) TEMP
         RETURN
         END
C
C
C
         SUBROUTINE MADD(A B C N1 N2 SCALE)
C
C THIS SUBROUTINE PERFORMS MATRIX (N1 X N2) ADDITION
C                  C A+(SCALE)*B
C
         IMPLICIT REAL*8 (A H O Z)
         DIMENSION A(N1 N2) B(N1 N2) C(N1 N2)
         DO 7902 I 1 N1
         DO 7902 J 1 N2
 7902 C(I J) A(I J)+SCALE*B(I J)
         RETURN
         END
C
C
C
C***************GETMN*********************************************
C
C  THIS SUBROUTINE EXTRACTS THE M AND N MATRICES AS DESCRIBED
C  IN THE MAIN PROGRAM INTRODUCTION   IT THEN PRINTS THESE
C
C*****************************************************************
         SUBROUTINE GETMN(N M P R NPLR NSUM BIGMIN BTMP BTMP1
        * RM RN)
         IMPLICIT REAL*8(A H O Z)
         INTEGER P R
         DIMENSION BIGMIN(NSUM NSUM) BTMP(NSUM M)
        *BTMP1(NSUM M) RM(NPLR M) RN(M M)
C  SET UP THE BTMP MATRIX
         DO 10 I-1 N
         DO 10 J-1 M
 10            BTMP(I J) 0 0
         DO 20 I 1 R
         DO 20 J 1 M
 20            BTMP(N+I J) 0 0
         DO 30 I 1 M
 30            BTMP(NPLR+I I) 1 0
C
   DO THE FINAL MATRIX MULTIPLY
C
         CALL MULT(BIGMIN BTMP BTMP1 NSUM NSUM M)
C  LOAD THE RM AND RN MATRICES
C
         DO 100 I=1 NPLR
         DO 100 J=1 M
 100           RM(I J) BTMP1(I J)
         DO 110 I 1 M
         DO 110 J 1 M
```

SETPNT Computer Program. (contd)

```
110       RN(I J) BTMP1(NPLR+I J)
C
C  PRINT THE RESULTS
C
      CALL MATPNT(RM NPLR M 'RM')
      CALL MATPNT(RN M M 'RN')
      RETURN
      END


      SUBROUTINE TFANAL(A B C M P R)
C
C  THIS SUBROUTINE FINDS THE MATRIX POLYNOMIAL AND RESIDUE POLE
C  REPRESENTATION OF THE COMPENSATOR DYNAMIC SYSTEM SHOWN BELOW
C
C              ZDOT = A * Z  +  B * YS
C                 U = C * Z
C
C  THE SUBROUTINE CALLS ARE BASED ON DOUG BERNARD'S
C  "CONTROLS PLAYPEN" LIBRARY
C
      IMPLICIT REAL*8(A H O Z)
      INTEGER DA(20) DB(20) DC(20) N(20) DN(20) DR(20) DENOM(20)
     *D(20) RR(20) LAMBDA(20) JMAT(20) NO(20) CLASS(64) RO(20)
      INTEGER P R
      COMPLEX*16 Z(10)
      DIMENSION A(R R) B(R P) C(M R) POLY(12)
      LODIM1=1
      LODIM2 1
      LODIM3 1
      L1DIM1 R
      L1DIM2 R
      L1DIM3 M
      L2DIM1=R
      L2DIM2=P
      L2DIM3=R
C
C  INITIALIZE DESCRIPTORS FOR DOUG'S PROGRAMS
C
      CALL INIT DESCRIPTOR(DA)
      CALL INIT DESCRIPTOR(DB)
      CALL INIT DESCRIPTOR(DC)
      CALL INIT DESCRIPTOR(N)
      CALL INIT DESCRIPTOR(DN)
      CALL INIT DESCRIPTOR(DR)
      CALL INIT_DESCRIPTOR(DENOM)
      CALL INIT DESCRIPTOR(D)
      CALL INIT DESCRIPTOR(RR)
      CALL INIT DESCRIPTOR(LAMBDA)
      CALL INIT DESCRIPTOR(JMAT)
      CALL INIT DESCRIPTOR(NO)
      CALL INIT DESCRIPTOR(RO)
      CALL COPY STATIC_TO DYNAMIC(A DA 1 R R LODIM1 L1DIM1 L2DIM1)
      CALL COPY STATIC_TO DYNAMIC(B DB 1 R P LODIM2 L1DIM2 L2DIM2)
      CALL COPY_STATIC_TO DYNAMIC(C DC 1 M R LODIM3 L1DIM3 L2DIM3)
      CALL PRT(DA 1  FALSE )
      CALL PRT(DB 1  FALSE )
      CALL PRT(DC 1  FALSE )
C
C  FIND THE MATRIX POLYNOMIAL FORM OF THE COMPENSATOR
C
      CALL FGH TO_MAT_POLY SEP(DA DB DC JMAT N D NO)
       WRITE(6 1200)
      CALL PRT(N 1  FALSE )
      CALL PRT(D 1  FALSE )
C
C  FIND THE POLES OF THE COMPENSATOR
C
       POLY(1) 1 0
       DO 200 I=1 R
200          POLY(I+1)=ELF(D 1 1 I)
       CALL ZPOLR(POLY R Z IERR)
       WRITE(6 1500)
       DO 210 II 1 R
210          WRITE(6 1100) Z(II)
C
C  FIND THE ZEROS OF EACH NUMERATOR T F
C  M IS # OF CONTROLS  P IS # OF MEAS
C
      DO 130 I 1 M
      DO 130 J 1 P
          KK 0
          K=1
160       IF (K EQ R+1) GO TO 150
          TMP ELF(N I J K)
          IF (DABS(TMP) LT 00001) GO TO 140
          KK KK+1
          POLY(KK)=TMP
140       K=K+1
          GO TO 160
150          WRITE(6 1400) (POLY(JJ) JJ 1 KK)
             CALL ZPOLR(POLY KK 1 Z IER)
             WRITE(6 1000) J I
             DO 170 II=1 KK 1
170               WRITE(6 1100) Z(II)
130   CONTINUE
C
C  FIND THE RESIDUE POLE FORM OF THE COMPENSATOR
C
      CALL MAT POLY TO_RESIDUE SEP(N D NO RR LAMBDA CLASS RO)
      WRITE(6 1300)
      CALL PRT(RR 1  FALSE )
      CALL PRT(LAMBDA 1  FALSE )
1000 FORMAT(' ' // 2X 'THE T F  ZEROS FROM INPUT ' I2 ' TO OUTPUT '
    * I2 / 2X 'REAL' 9X 'IMAGINARY')
1100 FORMAT(' ' 2X G12 5 3X G12 5)
1200 FORMAT(' ' // 2X 'THE MATRIX POLYNOMIAL REPRESENTATION OF
    * THE COMPENSATOR N(s)/d(s) IS' /)
1300 FORMAT(' ' // 2X 'THE RESIDUE FORM OF THE COMPENSATOR IS' /)
1400 FORMAT(' ' // 2X 'THE NUMERATOR COEFFICIENTS ARE' 10G12 4)
1500 FORMAT(' ' // 2X 'THE POLES OF THE COMPENSATOR ARE' /
    *'REAL' 9X 'IMAGINARY')
      RETURN
      END

c  ***************digit1******11/9/83**********************************
c
c  Written by Rick Holdridge
c
c  This subroutine accepts the analog form of a compensator in
c  block minimal form  then discretizes it using sample
c  time dtime  Then the discrete compensator is transformed
c  back into minimal form to speed up the calculations on the
c  real time computer
c
c  The form of the compensator is
c
c                 zdot = a * z + b * ys
```

SETPNT Computer Program. (contd)

```
          u    r * z

c   where     a     r x r  (compenator dynamics matrix)
              b     r x p  (compensator input dist  matrix)
              c     m x r  (compensator output dist  matrix)
              ys    p x 1  (measurement vector)
              u     m x 1  (control vector)
c
c**********************************************************************
      SUBROUTINE DIGITL(A B C ADIG BDIG CDIG DTIME R M P)
      implicit real*8 (a h o z)
      integer p r
      dimension a(r 1) b(r 1) c(m 1) adig(r 1) bdig(r 1)
     *cdig(m 1)
      do 50 i 1 r
      do 50 j 1 r
50       adig(i j) 0 0
c
c  Print out the input matrices for check purposes
c
       write(6 1000)
       call prnte(a r r r)
       write(6 1100)
       call prnte(b r r p)
       write(6 1200)
       call prnte(c m m r)
1000  format(// 1x 'The A matrix in digitl is')
1100  format(// 1x 'The B matrix in digitl is')
1200  format(// 1x 'The C matrix in digitl is')
c
c  Begin an outer loop which checks for 1x1 or 2x2 blocks
      i=1
100   if ((i+1) gt r) go to 200
      if ((dabs(a(i+1 i)) lt  0001) and (dabs(a(i i+1)) lt  0001))
     *go to 200
      go to 300
c
c  **200** This finds the part of the transition matrix
c          associated with a real mode
c
200   adig(i i)=exp(a(i i)*dtime)
      do 220 j=1 p
220       bdig(i j)=((exp(a(i i)*dtime) 1 0)/a(i i))*b(i j)
      do 230 j=1 m
230       cdig(j i) c(j i)
      i i+1
      if (i gt r) go to 400
      go to 100
c
c **300** This section finds the transision matrix for the
c         2x2 blocks
c
300   call phi(a(i+1,i) a(i+1 i+1) dtime adig(i i) adig(i i+1)
     * adig(i+1,i) adig(i+1 i+1) gam11 gam12 gam21 gam22)
c
c  fill the bdig matrix  I is the index of the phi(1 1) element
c  in the big phi matrix
c
      do 310 j=1.p
              bdig(i j)=gam12*b(i j)+gam11*b(i+1 j)
310           bdig(i+1 j)=gam22*b(i j)+gam21*b(i+1 j)
      do 320 j=1.m
              cdig(j.i)=c(j i)
320           cdig(j.i+1)=c(j.i+1)
      i i+2
      if (i gt r) go to 400
      go to 100
c
c  **400**  this last section minimalizes the matrix
c
 400  CONTINUE
c      write(6 1300)
c      call prnte(adig r r r)
c      write(6 1400)
c      call prnte(bdig r r p)
c      write(6 1500)
c      call prnte(cdig m m r)
1300  format(// 1x 'The compensator phi matrix is')
1400  format(// 1x 'The compensator gamma matrix is')
1500  format(// 1x 'The compensator output matrix is')
      call mincom(adig bdig cdig a b c r p m)
c
c  Now the a b c matrices have the digital form
c  of the compensator
c
      STOP
      end
c
c
c
       subroutine phi(a1 a2 T  phi11 phi12 phi21 phi22.
     1                          gam11 gam12 gam21 gam22)
c
c
comment This subroutine accepts a continuous 2 by 2 block in real
c        modal form and discretizes it
c
c        The continuous equations are
c
c        |z1dot|    |  0    1| |z1| + |0| * u1 + |1| * u2
c        |z2dot|    | a1   a2| |z2|   |1|        |0|
c
c        the discrete equations are
c
c        |z1|          |phi11 phi12| |z1|    + |gam11 gam12| |u1|
c        |z2|(k+1)     |phi21 phi22| |z2|(k)   |gam21 gam22| |u2|k
c
c        the sample period is T in units of seconds
c
c       note that the program returns unit solutions for the gamma matrix
c       For a real problem simply multiply the columns by the elements
c       of the continuous C matrix
c
        IMPLICIT REAL*8(A H O Z)
        REAL*8 NUM
        logical comp
        comp =  TRUE
c
c
comment check discriminant
c
c  these statements are needed since I enter a matrix of the
c  form
c          0    1
c         a1   a2
c                  not in the form used by Dan Rosenthal who
c                  wrote this subroutine
c
```

SETPNT Computer Program. (contd)

```
      a1   a1
      a2   a2
      disc   a2 * a2    4  * a1
      if ( disc  gt  0 ) comp   FALSE

      if ( comp  eq   true ) goto 1000

      a    0 5 * (a2 + sqrt(disc))
      b    0 5 * (a2   sqrt(disc))

      c    1
      d    a2
       call sub1(a b c d T x)
       phi11    x

     c    0
     d    1
       call sub1(a b c d T x)
       phi12 = x

     c = 0
     d =   a1
       call sub1(a b c d T x)
       phi21 - x

     c    1
     d    0
       call sub1(a b c d T x)
       phi22 = x

     num = 1
     call sub2(a b,T num x)
     gam11 = x

     gam21 = phi12

     gam12 = gam21 + a2 * gam11

     gam22 =    a1 * gam11

     goto 2000
1000 continue

comment two complex roots

      a = 0 5 * a2
      b = sqrt(a1   a*a)

      c = 1
      d = a2
       call sub3(a b c d T x)
       phi11 = x

     c = 0
     d = 1
       call sub3(a.b.c.d.T x)
       phi12 = x

     c = 0
     d =   a1
       call sub3(a.b c.d.T x)
       phi21 = x

      c = 1
      d   0
       call sub3(a b c d T x)
       phi22   x

      num   1
      call sub4(a b T num x)
      gam11   x

      gam21   phi12

      gam12   gam21 + a2 * gam11
      gam22      a1 * gam11

2000  RETURN
      end
      subroutine sub1(a b c d T x)
      IMPLICIT REAL*8(A H O Z)
               x    (d a*c)/(b a)* exp( a*T)
               x    x + (d  b*c)/(a  b)* exp( b*T)
      return
      end
      subroutine sub2(a b T num.x)
      IMPLICIT REAL*8(A H O Z)
      REAL*8 NUM
         c = 1
         d = a + b
         call sub1(a.b.c.d.T.temp)
         x = num/(a*b)* (1    temp)
      return
      end
      subroutine sub3(a b.c d.T x)
      IMPLICIT REAL*8(A H.O Z)
         x = c* cos(b*t) + (d   a*c)/b*sin(b*t)
         x = x * exp(  a*t)
       return
       end
       subroutine sub4( a.b T num x)
       IMPLICIT REAL*8(A H O Z)
       REAL*8 NUM
           c = 1 0
           d = 2 * a
           call sub3( a b.c d.T temp)
           x = num/(a*a +b*b)*( 1   temp)
       return
       end

      SUBROUTINE MINCOM(FMOD.GMOD HMOD.FMIN.GMIN.HMIN.NS.NC.NM)
C   *************MINCOM********11/9/83******************************************
C       WRITTEN BY RICK HOLDRIDGE
C
C       THIS PROGRAM CONVERTS A SYSTEM OF THE FORM
C               XDOT=FMOD*X + GMOD*U
C               Y HMOD*X
C       WHERE FMOD.GMOD AND HMOD ARE IN ARBITRARY FORM TO THE MINIMAL
C       FORM WHERE THE SYSTEM MATRICES TAKE THE FORM SHOWN BELOW
C
C             ! O   I                        !
C             ! A1 A2                        !
C             !       O   1       O          !
C             !       A1 A2                  !
C       FMIN  !             \                !   GMIN=NS X NC. FULL MATRIX
C             !               \              !
C             !       O         O   1        !
C             !                 A1 A2        !
```

SETPNT Computer Program. (contd)

```
C          :                                  \
C          :                                   A1  :
C          :  .     .     .     .                  :
C          :  O     1     .     .                  :
C          :  .     .     .     .   \              :
C    HMIN  :  .     .     O     1     \            :
C          :  .     .     .     .       \          :
C          :  .     .     .     .                  :
C          :  .     .     .     .             1    :
C
C    THE SUBROUTINE FINDS A TRANSFORMATION MATRIX TM SUCH THAT
C           FMIN  TMINV*FMOD*TM
C           GMIN TMINV*GMOD
C           HMIN HMOD*TM
C
C  ....................................................................
C
C  DO THE DIMENSIONING OF THE MATRICES
C
      IMPLICIT REAL*8(A H,O 2)
      DIMENSION FMOD(NS 1) HMOD(NM 1) GMOD(NS 1).
     *FMIN(NS.1) HMIN(NM.1) GMIN(NS 1)
      DIMENSION TMIN(20.20) TMININV(20 20) FTEMP(20 20)
C
C  Print out the input matrices for check purposes
C
       write(6.1001)
       call prntm(FMOD NS NS.NS)
       write(6 1101)
       call prntm(HMOD.NM.NM NS)
       write(6 1201)
       call prntm(GMOD.NS NS NC)
 1001 format(// 1x.'The FMOD matrix in MINCOM is')
 1101 format(// 1x.'The HMOD matrix in MINCOM is')
 1201 format(//.1x.'The GMOD matrix in MINCOM is')
C
C         CALCULATE THE TRANSFORMATION MATRIX  TMIN(NS NS)
C
      I=1
C
C   **********100***********
C   CHECK FOR REAL BLOCKS(NO OFF DIAGONAL TERMS)
C
  100 IF ((I+1) GT NS) GO TO 200
      IF ((DABS(FMOD(I+1 I)) LT  0001) AND
     *(DABS(FMOD(I.I+1)) LT  0001)) GO TO 200
      GO TO 300
C
C   **********200**********
C
  200 M=1
      RMM=ABS(HMOD(1 1))
      IF (NM EQ 1) GO TO 250
      DO 220 J=2.NM
         RMA=ABS(HMOD(J 1))
         IF (RMA LT RMM) GO TO 220
         RMM RMA
         M=J
C
C   **********220********
C
  220    CONTINUE
C
C   **********250********
C
  250 THIN(I 1) 1/HMOD(M 1)
      TMININV(I 1) HMOD(M 1)
      I I+1
      IF (I GT NS) GO TO 400
      GO TO 100
C
C   ***********300*********
C
  300 M 1
      RMM HMOD(1 1)**2+HMOD(1 I+1)**2
      IF (NM EQ 1) GO TO 350
      DO 320 J 2 NM
         RMA HMOD(J 1)**2+HMOD(J I+1)**2
         IF (RMA LT RMM) GO TO 320
         M J
         RMM RMA
C
C   *******320*******
C
  320 CONTINUE
C
C   **************350************
C
  350 A FMOD(I I)
      B FMOD(I.I+1)
      C FMOD(I+1 I)
      D FMOD(I+1 I+1)
      H1=+HMOD(M.I)
      H2-+HMOD(M.I+1)
      DENOM= B*H1**2+(A D)*H1*H2+C*H2**2
C
C  FOR DEBUGGING
C
C       WRITE(6.2000)A.B.C.D.H1.H2.DENOM.I.J.M NS.NC.NM
 2000 FORMAT(1X.'A='.F11 5 ' B='.F11 5 ' C ' F11 5.' D='.F11 5./.
     *1X.' H1='.F11 5 ' H2='.F11 5.' DENOM '.F11 5./.
     *1X.' I='.I1.'  J='.I1.'  M='.I1.'  NS-'.I1.'  NC='.I1.
     *'  NM='.I1)
      TMIN(I.I)=H2*(B*C A*D)/DENOM
      TMIN(I.I+1)=(A*H2 B*H1)/DENOM
      TMIN(I+1.I+1)-(C*H2 D*H1)/DENOM
      TMIN(I+1.I)= H1*(B*C-A*D)/DENOM
C
C       FIND THE INVERSE OF TMIN
C
      DENOM=TMIN(I+1.I+1)*TMIN(I.I)-TMIN(I I+1)*TMIN(I+1.I)
      TMININV(I.I)=TMIN(I+1.I+1)/DENOM
      TMININV(I+1.I+1)=TMIN(I.I)/DENOM
      TMININV(I I+1)= TMIN(I.I+1)/DENOM
      TMININV(I+1.I)= TMIN(I+1.I)/DENOM
      I=I+2
      IF (I GT NS) GO TO 400
      GO TO 100
C
C   *************400***********
C
C       TRANSFORM TO THE MINIMAL FORM
C
```

SETPNT Computer Program. (contd)

```
C        FMIN = TMININV*FMOD*TMIN
   400  DO 410 I=1,NS
        DO 410 J=1,NS
           FTEMP(I,J)=0.0
        DO 410 K=1,NS
          FTEMP(I,J)=FTEMP(I,J)+FMOD(I,K)*TMIN(K,J)
   410  CONTINUE
        DO 420 I=1,NS
        DO 420 J=1,NS
           FMIN(I,J)=0.0
        DO 420 K=1,NS
          FMIN(I,J)=FMIN(I,J)+TMININV(I,K)*FTEMP(K,J)
   420  CONTINUE
C        GMIN = TMININV*GMOD
        DO 430 I=1,NS
        DO 430 J=1,NC
           GMIN(I,J)=0.0
        DO 430 K=1,NS
          GMIN(I,J)=GMIN(I,J)+TMININV(I,K)*GMOD(K,J)
   430  CONTINUE
C        HMIN = HMOD*TMIN
        DO 440 I=1,NM
        DO 440 J=1,NS
           HMIN(I,J)=0.0
        DO 440 K=1,NS
          HMIN(I,J)=HMIN(I,J)+HMOD(I,K)*TMIN(K,J)
   440  CONTINUE
C        PRINT THE RESULTS
  1000 FORMAT('0',//,2X,'THE OUTPUT OF THE MINIMALIZING
      *SUBROUTINE IS ')
  1010 FORMAT('0',//,2X,'THE MINIMAL DISCRETE COMPENSATOR DYNAMICS
      *MATRIX  PHI  IS ')
  1020 FORMAT('0',//,2X,'THE MINIMAL DISCRETE COMPENSATOR INPUT
      *MATRIX  GAMMAMIN  IS ')
  1030 FORMAT('0',//,2X,'THE MINIMAL COMPENSATOR OUTPUT
      *MATRIX  HMIN  IS ')
  1050 FORMAT('0',//,2X,'THE TMIN MATRIX IS ')
  1060 FORMAT('0',//,2X,'THE TMININV MATRIX IS ')
  1070 FORMAT(10(2X,F11.5),/,2X,10(2X,F11.5))
       WRITE(6,1000)
       WRITE(6,1010)
       DO 600 I=1,NS
   600 WRITE(6,1070) (FMIN(I,J),J=1,NS)
       WRITE(6,1020)
       DO 610 I=1,NS
   610 WRITE(6,1070) (GMIN(I,J),J=1,NC)
       WRITE(6,1030)
       DO 620 I=1,NM
   620 WRITE(6,1070) (HMIN(I,J),J=1,NS)
       WRITE(6,1050)
       DO 640 I=1,NS
   640 WRITE(6,1070) (TMIN(I,J),J=1,NS)
       WRITE(6,1060)
       DO 650 I=1,NS
   650 WRITE(6,1070) (TMININV(I,J),J=1,NS)
       RETURN
       END
```

```
       SUBROUTINE PRNTM(X,MX,N,M)
       IMPLICIT REAL*8(A-H,O-Z)
       INTEGER COL1,COL10
       DIMENSION X(MX,1)
       COL1=1
       COL10=0
  1000 COL10=COL10+10
       IF (COL10.GT.M)COL10=M
       WRITE(6,1)(J,J=COL1,COL10)
       DO 450 I=1,N
   450 WRITE(6,3)I,(X(I,J),J=COL1,COL10)
       COL1=COL1+10
       IF (COL1.GT.M) RETURN
       GO TO 1000
  1    FORMAT(/,T9,10(9X,I3),/)
  3    FORMAT(T6,I3,T13,10(1X,G11.4))
       END
```

SETPNT Computer Program Example Data Set.

```
N M,P,R,SAMPLE TIME FOR SPHOV5 DAT
4 2,3,4, 5
F MATRIX( n X n )--U,W,Q,THETA
- 045,.036,0,-32.2
- 37,-2 02,176 0,0
 00191,-.0396,-2 98,0
0,0,1 0,0
G MATRIX( n X m )
0,1
-28 2,0
-11 0,0
0,0
HS MATRIX( p X n )--U,HDOT,THETA
1 0,0,0,0
0,-1 0,0,176 0
0,0,0,1 0
DSU MATRIX, PXM
0,0
0,0
0,0
 A MATRIX
0 0000E+00   1 000       0.0000E+00  0 0000E+00
-14 69       -6 072       0.0000E+00  0 0000E+00
0 0000E+00  0 0000E+00   -11 34       0.0000E+00
0 0000E+00  0 0000E+00   0 0000E+00   -1 816
 B MATRIX
-0 2781      -0 8863E-01   6 515
1 485       0 8725       -33 66
-0 1334E-01   -2 622        2 441
0 7711       -0 4641E-01   -10 10
 C MATRIX
0 000       -0 1000E-01 -0 1000E-01  0 3600E-02
-0 1978      -0 2163E-01  0 3446E-01   -1 000
D MATRIX, MXP
0,0,0
0,0,0
HD MATRIX, M X N+R,--U,HDOT,THETA
1,0,0,0,0,0,0,0,0
0,-1 0,0,176 0,0,0,0,0
LD MATRIX, M X M
0,0
0,0
```

SCALEM2 Computer Program. (contd)

```
1080 INPUT #1 TEMP$
1090 MAT INPUT #1 YSCALE
1100 INPUT #1 TEMP$
1110 MAT INPUT #1 CNTRLSCALE
1120 INPUT #1 TEMP$
1130 MAT INPUT #1 YDSCALE
1140 INPUT #1 TEMP$
1150 MAT INPUT #1 YMAX
1160 INPUT #1 TEMP$
1170 MAT INPUT #1 YIMAX
1180 REM
1190 REM  NOW FIND THE SCALING FOR THE COMPENSATOR STATES
1200 REM
1210 MAXELE=O
1220 MAXVEC=O
1230 FOR I=1 TO NR
1240        BVEC(I)=O'
1250 FOR J=1 TO NM
1260        BELEMENT=B(I,J)*BSCALE(J)*YMAX(J)
1270        BVEC(I)=BVEC(I)+BELEMENT
1280        IF ABS(BELEMENT)>MAXELE THEN MAXELE=ABS(BELEMENT)\IE=I\JE=J
1290 NEXT J
1300        IF ABS(BVEC(I))>MAXVEC THEN MAXVEC=ABS(BVEC(I))\IV=I
1310 NEXT I
1320 MAXELE2=LOG(ABS(MAXELE))/LOG(2)
1330 MAXVEC2=LOG(ABS(MAXVEC))/LOG(2)
1340 PRINT "POSSIBLE SCALING FOR Z(K) AND Z(K+1) BASED ON MAXIMUM ELEMENT OF B*Y
 IS"
1350 PRINT "            2^17/2^" MAXELE2
1360 PRINT "POSSIBLE SCALING FOR Z(K) AND Z(K+1) BASED ON MAXIMUM VECTOR OF B*Y
IS"
1370 PRINT "            2^17/2^" MAXVEC2
1380 INPUT "PLEASE TYPE IN THE DESIRED SCALING AS POWER OF 2",ZSCALE2
1390 ZSCALE=2^ZSCALE2
1400 REM
1410 REM  NOW DO THE CALCULATION OF THE MATRICES IN COMPUTER SCALED
1420 REM  VARIABLES
1430 REM
1440 MAXB=O
1450 MAXC=O
1460 MAXN=O
1470 MAXKI=O
1480 FOR I=1 TO NR
1490 FOR J=1 TO NM
1500        BCS(I,J)=B(I,J)*BSCALE(J)*ZSCALE/YSCALE(J)
1510        IF ABS(BCS(I,J))>MAXB THEN MAXB=ABS(BCS(I,J))
1520 NEXT J\NEXT I
1530 FOR I=1 TO NC
1540 FOR J=1 TO NR
1550        CCS(I,J)=C(I,J)*CSCALE(J)*CNTRLSCALE(I)/ZSCALE
1560        IF ABS(CCS(I,J))>MAXC THEN MAXC=ABS(CCS(I,J))
1570 NEXT J\NEXT I
1580 FOR I=1 TO NC
1590 FOR J=1 TO NC
1600        NCS(I,J)=N(I,J)*NSCALE(J)*CNTRLSCALE(I)/YDSCALE(J)
1610        IF ABS(NCS(I,J))>MAXN THEN MAXN=ABS(NCS(I,J))
1660 NEXT J\NEXT I
1670 MAXC2=LOG(MAXC)/LOG(2)
1680 MAXB2=LOG(MAXB)/LOG(2)
1690 MAXN2=LOG(MAXN)/LOG(2)
1700 PRINT
1710 PRINT "THE A MATRIX SCALE FACTOR WILL BE 2^15 OR 32768"
1720 PRINT "POSSIBLE SCALING FOR MATRIX B BASED ON MAX ELEMENT OF COMPUTER SCALE
D B IS"
1730 PRINT "            2^17/2^" MAXB2
1740 PRINT "POSSIBLE SCALING FOR MATRIX C BASED ON MAX ELEMENT OF COMPUTER SCALE
D C IS"
1750 PRINT "            2^17/2^" MAXC2
1760 PRINT "POSSIBLE SCALING FOR MATRIX N BASED ON MAX ELEMENT OF COMPUTER SCALE
D N IS"
1770 PRINT "            2^17/2^" MAXN2
1780 INPUT "PLEASE TYPE IN THE DESIRED SCALE FACTORS FOR MATRICES B C N " SCALEB
2,SCALEC2,SCALEN2
1790 REM
1800 REM NOW DO THE CALCULATION OF SCALE FACTORS FOR KX AND KXI
1810 REM NOTE   I ALSO DISCRETIZE THE KXI HERE BY MULTIPLYING BY TP(THE CYCLE TI
ME)
1820 REM          THE SCALED KXI MATRIX HAS A SAFETY FACTOR OF 5 TO AVOID OVERFLOW
S
1830 FOR I=1 TO NC
1840      SCALEKX(I)=INT(LOG(ABS(2^6/KX(I)))/LOG(2))
1850      SCALEKXI(I)=18
1855      SCALEKXD(I)=INT(LOG(ABS(2^12/KXD(I)))/LOG(2))
1860 NEXT I
1870 PRINT
1880 PRINT "THE KX SCALE FACTORS ARE"
1890 PRINT "  SCALEKX(1) " SCALEKX(1) "SCALEKX(2) " SCALEKX(2) "SCALEKX(3) " SCA
LEKX(3)
1900 PRINT "THE KXI SCALE FACTORS ARE"
1910 PRINT "  SCALEKXI(1) " SCALEKXI(1) "SCALEKXI(2) " SCALEKXI(2) "SCALEKXI(3)
" SCALEKXI(3)
1911 PRINT
1912 PRINT "THE KXD SCALE FACTORS ARE"
1914 PRINT "  SCALEKXD(1) " SCALEKXD(1),"SCALEKXD(2) " SCALEKXD(2) "SCALEKXD(3)
",SCALEKXD(3)
1920 REM
1930 REM  NOW DO THE FINAL SCALING OF THE MATRICES AND PRINT THEM
1940 REM
1950 FOR I=1 TO NR
1960         AFINAL(I)=A(I)*2^15
1970 NEXT I
1980 FOR I=1 TO NR
1990 FOR J=1 TO NM
2000         BFINAL(I,J)=BCS(I,J)*(2^SCALEB2)
2010 NEXT J\NEXT I
2020 FOR I=1 TO NC
2030 FOR J=1 TO NR
2040         CFINAL(I,J)=CCS(I,J)*(2^SCALEC2)
2050 NEXT J\NEXT I
2060 FOR I=1 TO NC
2070 FOR J=1 TO NC
2080         NFINAL(I,J)=NCS(I,J)*(2^SCALEN2)
2100 NEXT J\NEXT I
2110 FOR I=1 TO NC
2120         KXFINAL(I)=KX(I)*2^SCALEKX(I)
2130         KXIFINAL(I)=KXI(I)*2^SCALEKXI(I)
2135         KXDFINAL(I)=KXD(I)*2^SCALEKXD(I)
2140 NEXT I
2150 PRINT
2160 PRINT
2170 PRINT TAB(10),"THE PHYSICAL VARIABLE MATRICES ARE"
2180 PRINT TAB(10),"THE A MATRIX IS"
2190 FOR I=1 TO NR\ PRINT TAB(I*8),FNR(A(I)),\ NEXT I
2200 PRINT TAB(10),"THE B MATRIX IS"
2210 FOR I=1 TO NR\ FOR J=1 TO NM\ PRINT TAB(J*8),FNR(B(I,J)),\ NEXT J\ PRINT\ N
EXT I
2220 PRINT TAB(10),"THE C MATRIX IS"
2230 FOR I=1 TO NC\ FOR J=1 TO NR\ PRINT TAB(J*8),FNR(C(I,J)),\ NEXT J\ PRINT\ N
```

SCALEM2 Computer Program.

```
10 REM ******************** SCALEM2 BAS ********************
20 REM *
30 REM *    WRITTEN BY RICK HOLDRIDGE  10 FEB 84
40 REM *    UPDATED ON 5 JUNE 84 FOR THE HOVER AUTOPILOTS
45 REM *    MODIFIED TO RUN ON THE VAX ON 8 JUN 84
50 REM *
60 REM *   THIS PROGRAM IS USED TO SCALE THE A B C N KX KXI  AND KXD MATRICES
70 REM *   FOR USE IN THE CH 47 HELICOPTER CONTROL SYSTEM WHICH USES
80 REM *   THE SPERRY 1819A FIXED POINT DIGITAL COMPUTER
90 REM *
100 REM *          Z(K+1)    A * Z(K)  + B * YM(K)
110 REM *          U(K+1)    C*Z(K+1)  + N * [YD(K+1)  + KX * DELX(K+1)
120 REM *                              + KXI * INTDELX(K+1)  + KXD * XDOT(K+1)]
130 REM *                   WHERE    Z   COMPENSATOR STATES
140 REM *                           YM   MEASUREMENT VECTOR
150 REM *                           YD   DESIRED OUTPUT VECTOR
160 REM *                         DELX   POSITION ERROR VECTOR
170 REM *                      INTDELX   POSITION ERROR INTEGRAL VECTOR
175 REM *                         XDOT   INERTIAL VELOCITY
180 REM *                            A   COMPENSATOR DYNAMICS MATRIX
190 REM *                            B   COMPENSATOR OUTPUT DISTRIBUTION MATRIX
200 REM *                            C   COMPENSATOR OUTPUT DISTRIBUTION MATRIX
210 REM *                            N   FEEDFORWARD GAIN MATRIX TO ACHIEVE A
220 REM *                                DESIRED OUTPUT(INERTIAL VELOCITY)
230 REM *                           KX   PROPORTIONAL GAIN MATRIX ON POSITION
240 REM *                                ERROR (A DIAGONAL MATRIX)
250 REM *                          KXI   GAIN MATRIX ON THE INTEGRAL OF POSITION
260 REM *                                ERROR (A DIAGONAL MATRIX)
265 REM *                          KXD   GAIN MATRIX ON THE DERIVATIVE OF POSITION
267 REM *                                (INERTIAL VELOCITY) A DIAGONAL MATRIX
270 REM *
280 REM *   THE FOLLOWING DATA IS ENTERED INTO A FILE WHICH
290 REM *   THE PROGRAM WILL ASK THE NAME OF AND THEN READ THE
300 REM *   DATA FROM
310 REM *
320 REM *   NR NM NC TP        NR ORDER OF THE COMPENSATOR
330 REM *                      NM NUMBER OF MEASUREMENTS
340 REM *                      NC NUMBER OF CONTROLS AND DESIRED OUTPUTS
350 REM *                      TP CYCLE TIME IN SECONDS
360 REM *   A MATRIX(NR)       THE A MATRIX IS ENTERED AS A ROW CONTAINING
370 REM *                      THE MINIMAL NR NUMBER OF MATRIX ELEMENTS
380 REM *   B MATRIX(NR X NM) MEASUREMENTS DISTRIBUTION MATRIX ENTERED
390 REM *                      BY ROW
400 REM *   C MATRIX(NC X NR) COMPENSATOR STATE TO CONTROL DISTRIBUTION
410 REM *                      MATRIX ENTERED BY ROW
420 REM *   N MATRIX(NC X NC) COMMAND DISTRIBUTION MATRIX ENTERED BY ROW
430 REM *   KX MATRIX(NC)      POSITION ERROR TO CONTROL GAIN MATRIX
435 REM *                      ENTER DIAGONAL ELEMENTS
440 REM *   KXI MATRIX(NC)     POSITION ERROR INTEGRAL TO CONTROL GAIN MATRIX
450 REM *                      ENTER DIAGONAL ELEMENTS
455 REM *   KXD MATRIX(NC)     INERTIAL VELOCITY TO CONTROL GAIN MATRIX.
456 REM *                      ENTER DIAGONAL ELEMENTS
460 REM *   BSCALE(NR)         THE SCALING FACTORS ON SPECIFIC COLUMNS OF B
470 REM *                      THIS ALLOWS FOR CONVERSION FROM DEGREES TO
480 REM *                      RADIANS OR VICE VERSA
490 REM *   CSCALE(NR)         THE SCALING FACTORS ON SPECIFIC COLUMNS OF C
500 REM *                      NORMALLY. THESE TERMS ARE 1
510 REM *   NSCALE(NC)         THE SCALING FACTORS ON SPECIFIC COLUMNS OF N
520 REM *                      NORMALLY. THESE TERMS ARE 1
530 REM *   YSCALE(NM)         THE SCALING ON EACH MEASUREMENT   FOR INSTANCE
540 REM *                      IF Y(1) IS THETA AND THETA IS SCALED AS 500 BITS
550 REM *                      PER DEGREE  THEN YSCALE(1) 500
560 REM *   CNTRLSCALE(NC)     THE COMPUTER SCALING ON EACH CONTROL   FOR INSTANCE
570 REM *                      CONTROL(1) IS LONG CYCLIC AND IS SCALED AS 1024 BIT
S
580 REM *                      PER INCH  THEN CSCALE(1)=1024
590 REM *   YDSCALE(NC)        THE COMPUTER SCALING ON EACH DESIRED OUTPUT
600 REM *   YMAX(NM)           THE VECTOR OF MAX VALUES OF THE MEASUREMENTS
610 REM *   YDMAX(NC)          THE VECTOR OF MAX VALUES OF THE DESIRED OUTPUTS
620 REM *
630 REM ************************************************************
640 REM
650 REM  READ THE DATA FROM A FILE
660 REM
670 DEF FNR(S)  (1000000*S+ 5)/1000000
680 PRINT "WELCOME TO SCALEM"
690 INPUT "PLEASE TYPE IN THE NAME OF THE FILE CONTAINING THE INPUT DATA" FILENA
ME$
700 OPEN FILENAME$ FOR INPUT AS #1
710 INPUT #1  TEMP$
720 INPUT #1  NR NM NC  TP
730 DIM A(NR) AFINAL(NR) B(NR NM) BCS(NR NM) BFINAL(NR NM) BTEMP(NR NM)
740 DIM C(NC NR) CCS(NC NR) CFINAL(NC NR) CTEMP(NC NR) N(NC NC) NCS(NC NC)
750 DIM NFINAL(NC NC) NTEMP(NC NC) KX(NC) KXFINAL(NC)
755 DIM KXI(NC) KXIFINAL(NC) KXD(NC) KXDFINAL(NC)
760 DIM KITEMP(NC NC) BSCALE(NM) CSCALE(NR) NSCALE(NC) YSCALE(NM)
770 DIM CNTRLSCALE(NC) YDSCALE(NC) YMAX(NM) YDMAX(NC) BVEC(NR)
775 DIM SCALEKX(NC) SCALEKXI(NC) SCALEKXD(NC)
780 INPUT #1 TEMP$
790 MAT INPUT #1 A
800 PRINT TAB(10),"THE A MATRIX IS"
810 FOR I-1 TO NR\ PRINT TAB(I*8) FNR(A(I)) \ NEXT I
820 INPUT #1 TEMP$
830 MAT INPUT #1 B
840 PRINT TAB(10) "THE B MATRIX IS"
850 FOR I 1 TO NR\ FOR J 1 TO NM\ PRINT TAB(J*8) FNR(B(I J)) \ NEXT J\ PRINT\ NE
XT I
860 INPUT #1,TEMP$
870 MAT INPUT #1 C
880 PRINT TAB(10) "THE C MATRIX IS"
890 FOR I 1 TO NC\ FOR J 1 TO NR\ PRINT TAB(J*8) FNR(C(I J)) \ NEXT J\ PRINT\ NE
XT I
900 INPUT #1,TEMP$
910 MAT INPUT #1 N
920 PRINT TAB(10) "THE N MATRIX IS"
930 FOR I 1 TO NC\ FOR J 1 TO NC\ PRINT TAB(J*8) FNR(N(I J)) \ NEXT J\ PRINT\ NE
XT I
940 INPUT #1 TEMP$
950 MAT INPUT #1 KX
960 PRINT TAB(10) "THE KX MATRIX IS"
970 FOR I 1 TO NC\ PRINT TAB(I*8),FNR(KX(I)) \ NEXT I
980 INPUT #1,TEMP$
990 MAT INPUT #1 KXI
1000 PRINT TAB(10),"THE KXI MATRIX IS"
1010 FOR I=1 TO NC\ PRINT TAB(I*8),FNR(KXI(I)) \ NEXT I
1012 INPUT #1 TEMP$
1014 MAT INPUT #1,KXD
1016 PRINT TAB(10),"THE KXD MATRIX IS"
1018 FOR I 1 TO NC\ PRINT TAB(I*8) FNR(KXD(I)) \ NEXT I
1020 INPUT #1 TEMP$
1030 MAT INPUT #1 BSCALE
1040 INPUT #1,TEMP$
1050 MAT INPUT #1 CSCALE
1060 INPUT #1,TEMP$
1070 MAT INPUT #1 NSCALE
```

SCALEM1 Computer Program. (contd)

```
1130 INPUT "PLEASE TYPE IN THE DESIRED SCALING AS POWER OF 2" ZSCALE2
1140 ZSCALE = 2 ^ ZSCALE2
1150 REM
1160 REM  NOW DO THE CALCULATION OF THE MATRICES IN COMPUTER SCALED
1170 REM  VARIABLES
1180 REM
1190 MAXB=0
1200 MAXC=0
1210 MAXN=0
1220 MAXK1=0
1230 FOR I=1 TO NB
1240     FOR J=1 TO NN
1250         BCS(I,J)=B(I,J)*SCALE(J)/ZSCALE/YSCALE(J)
1260         IF ABS(BCS(I,J))>MAXB THEN MAXB=ABS(BCS(I,J))
1270 NEXT J\NEXT I
1280 FOR I=1 TO NC
1290     FOR J=1 TO NN
1300         CCS(I,J)=C(I,J)*SCALE(J)*CNTRLSCALE(J)/ZSCALE
1310         IF ABS(CCS(I,J))>MAXC THEN MAXC=ABS(CCS(I,J))
1320 NEXT J\NEXT I
1330 FOR I=1 TO NC
1340     FOR J=1 TO NC
1350         NCS(I,J)=N(I,J)*NSCALE(J)*CNTRLSCALE(J)/YSCALE(J)
1360         IF ABS(NCS(I,J))>MAXN THEN MAXN=ABS(NCS(I,J))
1370 REM MULTIPLY THE K1 MATRIX BY THE CYCLE TIME FOR DISCRETIZATION
1380     KI(I,J)=TP*KI(I,J)
1390     KICS(I,J)=KI(I,J)*KISCALE(J)*CNTRLSCALE(J)/YSCALE(J)
1400     IF ABS(KICS(I,J))>MAXK1 THEN MAXK1=ABS(KICS(I,J))
1410 NEXT J\NEXT I
1420 MAXC2=LOG(MAXC)/LOG(2)
1430 MAXB2=LOG(MAXB)/LOG(2)
1440 MAXN2=LOG(MAXN)/LOG(2)
1450 REM THE NE NEED 15 INCLUDED SINCE THE INTEGAL CONTROL IS A SUMMATION
1460 REM AND WE NEED A SAFETY FACTOR(OF 32) TO AVOID OVERFLOWS
1470 MAXK12=LOG(MAXK1)/LOG(2)+5
1480 PRINT
1490 PRINT "THE A MATRIX SCALE FACTOR WILL BE 2 15 OR 32768"
1500 PRINT "POSSIBLE SCALING FOR MATRIX B BASED ON MAX ELEMENT OF COMPUTER SCALE
D B IS"
1510 PRINT "                2 17/2 " MAXB2
1520 PRINT "POSSIBLE SCALING FOR MATRIX C BASED ON MAX ELEMENT OF COMPUTER SCALE
D C IS"
1530 PRINT "                2 17/2 " MAXC2
1540 PRINT "POSSIBLE SCALING FOR MATRIX N BASED ON MAX ELEMENT OF COMPUTER SCALE
D N IS"
1550 PRINT "                2 17/2 " MAXN2
1560 PRINT "POSSIBLE SCALING FOR MATRIX TP*K1 BASED ON MAX ELEMENT OF COMPUTER"
1570 PRINT "SCALED TP*K1 INCLUDING A SAFETY FACTOR OF 2^5 IS"
1580 PRINT "                2 17/2 " MAXK12
1590 INPUT "PLEASE TYPE IN THE DESIRED SCALE FACTORS FOR MATRICES B C N TP*K1" S
CALE2 SCALEC2 SCALEN2 SCALEK12
1600 REM
1610 REM  NOW DO THE FINAL SCALING OF THE MATRICES AND PRINT THEM
1620 REM
1630 FOR I=1 TO NB
1640     AFINAL(I)=A(I)*2^15
1650 NEXT I
1660 FOR I=1 TO NB
1670     BFINAL(I,J)=BCS(I,J)*(2^SCALEB2)
1680 NEXT J\NEXT I
1690 FOR I=1 TO NC
1700     FOR J=1 TO NN
1710         CFINAL(I,J)=CCS(I,J)*(2^SCALEC2)
1720 NEXT J\NEXT I
1730 FOR I=1 TO NC
1740     FOR J=1 TO NN
1750         NFINAL(I,J)=NCS(I,J)*(2^SCALEN2)
1760 NEXT J\NEXT I
1770 FOR I=1 TO NC
1780     FOR J=1 TO NC
1790         KIFINAL(I,J)=KICS(I,J)*(2^SCALEK12)
1800 NEXT J\NEXT I
1810 PRINT
1820 PRINT
1830 PRINT TAB(10) "THE PHYSICAL VARIABLE MATRICES ARE"
1840 PRINT TAB(10) "THE A MATRIX IS"
1850 PRINT TAB(10) "THE B MATRIX IS"
1860 FOR I=1 TO NB\ PRINT TAB(I*8) FNR(A(I)) \ NEXT I
1870 PRINT
1880 FOR I=1 TO NB\ PRINT TAB(I*8) FNR(B(I,J)) \ NEXT J\ PRINT\ N
EXT I
1890 PRINT TAB(10) "THE C MATRIX IS"
1900 FOR I=1 TO NC\ FOR J=1 TO NN\ PRINT TAB(J*8) FNR(C(I,J)) \ NEXT J\ PRINT\ N
EXT I
1910 PRINT TAB(10) "THE N MATRIX IS"
1920 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8) FNR(N(I,J)) \ NEXT J\ PRINT\ N
EXT I
1930 PRINT TAB(10) "THE TP*K1 MATRIX IS"
1940 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8) FNR(KI(I,J)) \ NEXT J\ PRINT\
NEXT I
1950 PRINT
1960 PRINT TAB(10) "THE COMPUTER SCALED VARIABLE MATICES ARE"
1970 PRINT TAB(10) "THE COMPUTER SCALED B MATRIX IS"
1980 FOR I=1 TO NB\ FOR J=1 TO NN\ PRINT TAB(J*8) FNR(BCS(I,J)) \ NEXT J\ PRINT\
NEXT I
1990 PRINT TAB(10) "THE COMPUTER SCALED C MATRIX IS"
2000 FOR I=1 TO NC\ FOR J=1 TO NN\ PRINT TAB(J*8) FNR(CCS(I,J)) \ NEXT J\ PRINT\
NEXT I
2010 PRINT TAB(10) "THE COMPUTER SCALED N MATRIX IS"
2020 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8) FNR(NCS(I,J)) \ NEXT J\ PRINT\
NEXT I
2030 PRINT TAB(10) "THE COMPUTER SCALED TP*K1 MATRIX IS"
2040 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8) FNR(KICS(I,J)) \ NEXT J\ PRINT
\ NEXT I
2050 PRINT
2060 PRINT TAB(10) "THE COMPLETELY SCALED MATRICES ARE"
2070 PRINT TAB(10) "THE AFINAL MATRIX IS"
2080 FOR I=1 TO NB\ PRINT TAB(I*8) FNR(AFINAL(I)) \ NEXT I
2090 PRINT TAB(10) "THE BFINAL MATRIX IS"
2100 FOR I=1 TO NB\ PRINT TAB(J*8) FNR(BFINAL(I,J)) \ NEXT J\ PRI
NT\ NEXT I
2110 PRINT TAB(10) "THE CFINAL MATRIX IS"
2120 FOR I=1 TO NC\ FOR J=1 TO NN\ PRINT TAB(J*8) FNR(CFINAL(I,J)) \ NEXT J\ PRI
NT\ NEXT I
2130 PRINT TAB(10) "THE NFINAL MATRIX IS"
2140 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8) FNR(NFINAL(I,J)) \ NEXT J\ PRI
NT\ NEXT I
2150 PRINT TAB(10) "THE (TP*K1)FINAL MATRIX IS"
2160 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8) FNR(KIFINAL(I,J)) \ NEXT J\ PR
INT\ NEXT I
2170 PRINT
2180 PRINT "THE A MATRIX REQUIRES A LATA OF 3 THEN STRAU IN 1619 ASSEMBLY"
2190 PRINT "THE B MATRIX REQUIRES A LATA OF 18 SCALEB2 "THEN STRAU"
2200 PRINT "THE C MATRIX REQUIRES A LATA OF 18 SCALEC2 "THEN STRAU"
2210 PRINT "THE N MATRIX REQUIRES A LATA OF 18 SCALEN2 "THEN STRAU"
2220 PRINT "THE K1 MATRIX REQUIRES A LATA OF 18 SCALEK12 "THEN STRAU"
2230 END
```

SCALEM1 Computer Program.

```
10 REM ******************** SCALEM1 ***********************
20 REM *
30 REM *   WRITTEN BY RICK HOLDRIDGE. 10 FEB 84
35 REM *   MODIFIED TO RUN ON THE VAX. 8 JUN 84
40 REM *
50 REM *  THIS PROGRAM IS USED TO SCALE THE A.B.C.N MATRICES FOR
60 REM *  USE IN THE CH 47 HELICOPTER CONTROL SYSTEM WHICH USES
70 REM *  THE SPERRY 1819A FIXED POINT DIGITAL COMPUTER
80 REM *
90 REM *  THE FOLLOWING DATA IS ENTERED INTO A FILE WHICH
100 REM *  THE PROGRAM WILL ASK THE NAME OF AND THEN READ THE
110 REM *  DATA FROM
120 REM *
130 REM *  NR NM NC TP        NR ORDER OF THE COMPENSATOR
140 REM *                     NM NUMBER OF MEASUREMENTS
150 REM *                     NC NUMBER OF CONTROLS AND DESIRED OUTPUTS
160 REM *                     TP CYCLE TIME IN SECONDS
170 REM *  A MATRIX(NR)       THE A MATRIX IS ENTERED AS A ROW CONTAINING
180 REM *                     THE MINIMAL NR NUMBER OF MATRIX ELEMENTS
190 REM *  B MATRIX(NR X NM) MEASUREMENTS DISTRIBUTION MATRIX ENTERED
200 REM *                      BY ROW
210 REM *  C MATRIX(NC X NR) COMPENSATOR STATE TO CONTROL DISTRIBUTION
220 REM *                      MATRIX ENTERED BY ROW
230 REM *  N MATRIX(NC X NC) COMMAND DISTRIBUTION MATRIX ENTERED BY ROW
240 REM *  KI MATRIX(NC X NC)INTEGRAL GAIN MATRIX
250 REM *  BSCALE(NR)          THE SCALING FACTORS ON SPECIFIC COLUMNS OF B
260 REM *                      THIS ALLOWS FOR CONVERSION FROM DEGREES TO
270 REM *                      RADIANS OR VICE VERSA
280 REM *  CSCALE(NR)          THE SCALING FACTORS ON SPECIFIC COLUMNS OF C
290 REM *                      NORMALLY. THESE TERMS ARE 1
300 REM *  NSCALE(NC)          THE SCALING FACTORS ON SPECIFIC COLUMNS OF N
310 REM *                      NORMALLY. THESE TERMS ARE 1
320 REM *  KISCALE(NC)         THE SCALING FACTORS ON SPECIFIC COLUMNS OF KI
330 REM *                      NORMALLY. THESE TERMS ARE 1
340 REM *  YSCALE(NM)          THE SCALING ON EACH MEASUREMENT  FOR INSTANCE.
350 REM *                      IF Y(1) IS THETA AND THETA IS SCALED AS 500 BITS
360 REM *                      PER DEGREE. THEN YSCALE(1)=500
370 REM *  CNTRLSCALE(NC)      THE COMPUTER SCALING ON EACH CONTROL  FOR INSTANCE

380 REM *                      CONTROL(1) IS LONG CYCLIC AND IS SCALED AS 1024 BIT
S
390 REM *                      PER INCH. THEN CSCALE(1)=1024
400 REM *  YDSCALE(NC)         THE COMPUTER SCALING ON EACH DESIRED OUTPUT
410 REM *  YMAX(NM)            THE VECTOR OF MAX VALUES OF THE MEASUREMENTS
420 REM *  YDMAX(NC)           THE VECTOR OF MAX VALUES OF THE DESIRED OUTPUTS
430 REM *
440 REM ****************************************************************
450 REM
460 REM  READ THE DATA FROM A FILE
470 REM
480 DEF FNR(S)=(1000000*S+.5)/1000000
490 PRINT "WELCOME TO SCALEM1"
500 INPUT "PLEASE TYPE IN THE NAME OF THE FILE CONTAINING THE INPUT DATA".FILENA
ME$
510 OPEN FILENAME$ FOR INPUT AS #1
520 INPUT #1. TEMP$
530 INPUT #1. NR.NM.NC.TP
540 DIM A(NR).AFINAL(NR).B(NR.NM).BCS(NR.NM).BFINAL(NR.NM).BTEMP(NR.NM)
550 DIM C(NC.NR).CCS(NC.NR).CFINAL(NC.NR).CTEMP(NC.NR).N(NC.NC).NCS(NC.NC)
560 DIM NFINAL(NC.NC).NTEMP(NC.NC).KI(NC.NC).KICS(NC.NC).KIFINAL(NC.NC)
570 DIM KITEMP(NC.NC).BSCALE(NM).CSCALE(NR).NSCALE(NC).KISCALE(NC).YSCALE(NM)
580 DIM CNTRLSCALE(NC).YDSCALE(NC).YMAX(NM).YDMAX(NC).BVEC(NR)
590 INPUT #1.TEMP$
600 MAT INPUT #1.A
610 PRINT TAB(10)."THE A MATRIX IS"
620 FOR I=1 TO NR\ PRINT TAB(I*8).FNR(A(I)) \ NEXT I
630 INPUT #1.TEMP$
640 MAT INPUT #1.B
650 PRINT TAB(10)."THE B MATRIX IS"
660 FOR I=1 TO NR\ FOR J=1 TO NM\ PRINT TAB(J*8).FNR(B(I.J)) \ NEXT J\ PRINT\ NE
XT I
670 INPUT #1.TEMP$
680 MAT INPUT #1.C
690 PRINT TAB(10)."THE C MATRIX IS"
700 FOR I=1 TO NC\ FOR J=1 TO NR\ PRINT TAB(J*8).FNR(C(I.J)) \ NEXT J\ PRINT\ NE
XT I
710 INPUT #1.TEMP$
720 MAT INPUT #1.N
730 PRINT TAB(10)."THE N MATRIX IS"
740 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8).FNR(N(I.J)) \ NEXT J\ PRINT\ NE
XT I
750 INPUT #1.TEMP$
760 MAT INPUT #1.KI
770 PRINT TAB(10)."THE KI MATRIX IS"
780 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8).FNR(KI(I.J)) \ NEXT J\ PRINT\ N
EXT I
790 INPUT #1.TEMP$
800 MAT INPUT #1.BSCALE
810 INPUT #1.TEMP$
820 MAT INPUT #1.CSCALE
830 INPUT #1.TEMP$
840 MAT INPUT #1.NSCALE
850 INPUT #1.TEMP$
860 MAT INPUT #1.KISCALE
870 INPUT #1.TEMP$
880 MAT INPUT #1.YSCALE
890 INPUT #1.TEMP$
900 MAT INPUT #1.CNTRLSCALE
910 INPUT #1.TEMP$
920 MAT INPUT #1.YDSCALE
930 INPUT #1.TEMP$
940 MAT INPUT #1.YMAX
950 INPUT #1.TEMP$
960 MAT INPUT #1.YDMAX
970 REM
980 REM  NOW FIND THE SCALING FOR THE COMPENSATOR STATES
990 REM
1000 MAXELE=0
1010 MAXVEC=0
1020 FOR I=1 TO NR
1030        BVEC(I)=0'
1040 FOR J=1 TO NM
1050        BELEMENT=B(I.J)*BSCALE(J)*YMAX(J)
1060        BVEC(I)=BVEC(I)+BELEMENT
1070        IF ABS(BELEMENT)>MAXELE THEN MAXELE=ABS(BELEMENT)\IE=I\JE=J
1080 NEXT J
1090        IF ABS(BVEC(I))>MAXVEC THEN MAXVEC=ABS(BVEC(I))\IV=I
1100 NEXT I
1110 MAXELE2=LOG(ABS(MAXELE))/LOG(2)
1120 MAXVEC2=LOG(ABS(MAXVEC))/LOG(2)
1130 PRINT "POSSIBLE SCALING FOR Z(K) AND Z(K+1) BASED ON MAXIMUM ELEMENT OF B*Y
 IS"
1140 PRINT "           2^17/2^".MAXELE2
1150 PRINT "POSSIBLE SCALING FOR Z(K) AND Z(K+1) BASED ON MAXIMUM VECTOR OF B*Y
IS"
1160 PRINT "           2^17/2^".MAXVEC2
```

Appendix N.

SCALEM1 and SCALEM2 Computer Programs

This appendix lists the computer programs for doing the fixed point scaling described in Appendix E. Also included are two example sets of data. These programs are written in VAX BASIC. The SCALEM1 computer program does the scaling for the longitudinal CAS controller. The SCALEM2 computer program does the scaling for the hover controller. The description of required data is at the beginning of each program. The SCALEM1 data is for the 6^{th} order compensator with integral control. The SCALEM2 data is for the final hover controller.

SIMPLOT Example Data File.

```
4,8,0,.1,15.0,0,0
18,22,20,21,16,17,18,22
'NAVION WITH FULL ORDER COMP HDOT COM = 10 FT/SEC  '
'FORWARD VEL(FT/SEC)  '
'CLIMB RATE(FT/SEC)  '
' PITCH RATE(DEG/SEC) '
' THETA (DEG)  '
' ELEVATOR(DEGREES)  '
' THROTTLE(FT/SEC**2) '
'FORWARD VEL(FT/SEC)  '
'CLIMB RATE(FT/SEC)  '
0,0,0,0,0,0,10,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

SIMPLOT Computer Program. (contd)

```
      DO 3100 I=1,NPNTS
              YARAY(I)=DAT(NCNT+1,I)
              IF (YARAY(I) GT YMAX) YMAX=YARAY(I)
              IF (YARAY(I) LT YMIN) YMIN=YARAY(I)
3100  CONTINUE
      CALL PHYSOR(X3ORG(NCRVE),Y3ORG(NCRVE))
      CALL AREA2D(6.0,2.7)
      CALL XNAME('TIME(sec)',9)
      CALL YNAME(%REF(PLTIT(NCNT)),20)
      CALL CROSS
      CALL GRAF(0.,'SCALE',TIME(NPNTS),YMIN,'SCALE',YMAX)
      CALL CURVE(TIME,YARAY,NPNTS,0)
      CALL ENDGR(NPCNT)
4400  CONTINUE
      CALL ENDPL(NPCNT)
4500  CONTINUE
      GO TO 5000

C THIS IS THE PART THAT DOES 4 PLOTS/PAGE

4000 DO 4500 NPCNT=1,NPAGE
      CALL PAGE(11.0,8.5)
      CALL PHYSOR(1.,.1)
      IF (NPLTR.EQ.0) CALL HWROT('MOVIE')
      CALL NOBRDR
      CALL AREA2D(9.0,6.0)
      CALL HEADIN(%REF(GRTIT),50,2.1)
      CALL ENDGR(NPCNT)

C NOW START THE LOOP FOR PLOTTING THE 4 CURVES

      DO 4400 NCRVE=1,4

C NCNT IS THE COLUMN OF THE VECTOR IN DAT BEING PLOTTED

      NCNT=(4*NPCNT-4)+NCRVE
      YMAX=DAT(NCNT+1,1)
      YMIN=YMAX
      DO 4100 I=1,NPNTS
              YARAY(I)=DAT(NCNT+1,I)
              IF (YARAY(I) GT YMAX) YMAX=YARAY(I)
              IF (YARAY(I) LT YMIN) YMIN=YARAY(I)
4100  CONTINUE
      CALL PHYSOR(X4ORG(NCRVE),Y4ORG(NCRVE))
      CALL AREA2D(4.1,2.7)
      CALL XNAME('TIME(sec)',9)
      CALL YNAME(%REF(PLTIT(NCNT)),20)
      CALL CROSS
      CALL GRAF(0.,'SCALE',TIME(NPNTS),YMIN,'SCALE',YMAX)
      CALL CURVE(TIME,YARAY,NPNTS,0)
      CALL ENDGR(NPCNT)
4400  CONTINUE
      CALL ENDPL(NPCNT)
4500  CONTINUE
      GO TO 5000
5000  CALL DONEPL
      RETURN
      END
```

SIMPLOT Computer Program. (contd)

```
C       NPLTR     =0  FOR OUTPUT TO VECTRI PLV WHICH CAN BE SENT TO
C                     THE VERSATEC FOR HARD COPIES
C                 =1  FOR THE PLOTS TO BE DISPLAYED ON A VT125
C
C       NPNTS THE TOTAL NUMBER OF DATA POINTS
C
C       CRTIT A CHARACTER*50 VARIABLE CONTAINING THE TITLE TO BE
C             ON ALL PAGES OF OUTPUT
C
C       PLTIT A CHARACTER*20 ARRAY OF 12 TITLES OF THE Y AXES OF
C             THE PLOTS
C
C
C
C*****************************************************************
        SUBROUTINE PLOTEM(DAT,NTYPE,NPLT,NPLTR,NPNTS,CRTIT,PLTIT)
        IMPLICIT REAL(A-H,O-Z)
        CHARACTER*50 CRTIT
        CHARACTER*20 PLTIT(12)
        DIMENSION DAT(13,5000),TIME(5000),YARAY(5000),X3ORG(3),Y3ORG(3)
        DIMENSION X4ORG(4),Y4ORG(4),NPARM(12)
        DATA X3ORG/1.25,1.25,1.25/
        DATA Y3ORG/7.1,4.0,.9/
        DATA X4ORG/1.25,6.25,1.25,6.25/
        DATA Y4ORG/4.25,4.25,0.1,0/
C
C STORE THE TIME POINT IN THE TIME ARRAY
C
        DO 10 I=1,NPNTS
              TIME(I)=DAT(1,I)
 10     CONTINUE
        NPAGE=NPLT/NTYPE
C
C SET UP VERSATEC PLOTTER OR USE VT125
C
        IF (NPLTR.EQ.0) GO TO 20
        CALL REGIS(1,0)
        GO TO 30
 20     CALL VRSTEC(0,0,0)
 30     CONTINUE
C
C NOW DETERMINE THE TYPE OF PAGE FORMAT
C
        IF (NTYPE.EQ.1) GO TO 1000
        IF (NTYPE.EQ.3) GO TO 3000
        IF (NTYPE.EQ.4) GO TO 4000
C       WRITE(6,100)
C 100   FORMAT(1X,'NTYPE WAS NOT 1,3, OR 4')
        GO TO 5000
C
C DO THE CASE OF ONLY ONE PLOT PER PAGE
C
 1000   DO 1500 NCNT=1,NPAGE
C
C LOAD THE YARAY WITH THE DATA TO BE PLOTTED AND
C ALSO FIND THE MAX VALUE OF Y
C
        YMAX=DAT(NCNT+1,1)
        YMIN=DAT(NCNT+1,1)
        DO 1100 I=1,NPNTS
              YARAY(I)=DAT(NCNT+1,I)
              IF (YARAY(I).GT.YMAX) YMAX=YARAY(I)
              IF (YARAY(I).LT.YMIN) YMIN=YARAY(I)
 1100   CONTINUE
        WRITE(6,*)YMAX
C
C USE HORIZONTAL PAGE FORMAT FOR VERSATEC
C OR VERTICLE FOR THE VT125
C
        CALL PAGE(11.,8.5)
        CALL PHYSOR(1.,1.)
        IF (NPLTR.EQ.0) CALL HWROT('MOVIE')
C
C SUPPRESS PAGE BORDER OUTLINE
C
        CALL NOBRDR
C
C SET UP PLOTTING AREA
C
        CALL AREA2D(9.0,6.0)
C
C PUT THE TITLE ON THE GRAPH
C
        CALL HEADIN(%REF(CRTIT),50,2,1)
C
C THE X AXIS WITH TIME IN SECONDS
C
        CALL XNAME('TIME(sec)',9)
C
C LABEL THE ORDINATE WITH THE VARIABLE NAME
C
        CALL YNAME(%REF(PLTIT(NPAGE)),20)
C
C SET UP THE GRAPH TYPE USING SELF SCALING
C AND PUT THE AXIS AT Y=0
C
        CALL CROSS
        CALL GRAF(0.,'SCALE',TIME(NPNTS),YMIN,'SCALE',YMAX)
C
C PLOT THE DATA
C
        CALL CURVE(TIME,YARAY,NPNTS,0)
C
C END ONE PAGE OF ONE PLOT
C
        CALL ENDPL(NCNT)
 1500   CONTINUE
        GO TO 5000
C
C THIS IS THE PART THAT DOES 3 PLOTS/PAGE
C
 3000 DO 3500 NPCNT=1,NPAGE
        CALL PAGE(8.5,11.)
        CALL PHYSOR(1.,1.)
        CALL NOBRDR
        CALL AREA2D(6.0,8.5)
        CALL HEADIN(%REF(CRTIT),50,2,1)
        CALL ENDGR(NPCNT)
C
C NOW START THE LOOP FOR PLOTTING THE 3 CURVES
C
        DO 3400 NCRVE=1,3
C
C NCNT IS THE COLUMN OF THE VECTOR IN DAT BEING PLOTTED
C
        NCNT=(3*NPCNT-3)+NCRVE
        YMAX=DAT(NCNT+1,1)
        YMIN=YMAX
```

SIMPLOT Computer Program. (contd)

```
      continue
      do 16 i=1,nc
      do 16 k=1,ns
      x=y(k)
      y(nc+i)=y(ns+i)+CTB(i,k)*x
16    continue
      do 17 i=1,nc
      u(i)=y(ns+i)
17    continue
      do 18 i=1,no
      do 18 k=1,ns
      y(ns+nc+i)=y(ns+nc+i)+hmat(i,k)*y(k)
18    continue
      do 19 i=1,no
      do 19 k=1,nc
      y(ns+nc+i)=y(ns+nc+i)+dmat(i,k)*u(k)
19    continue
      if(nd.eq.0)go to 25
      do 25 i=1,no
      do 25 k=1,nd
      y(ns+nc+i)=y(ns+nc+i)+dvmat(i,k)*(wO(k)+dist(k))
25    continue
      do 300 i=1,no
      opt(i)=y(ns+nc+i)
      av(i,k)=y(ns+nc+i)
300   continue
      do 301 i=1,nc
      opt(no+i)=u(i)
301   continue
      if(DABS(t-(dt*k4+t0)).GE. .00001) GO TO 1201
C     WRITE DATA TO THE DAT ARRAY
      DAT(1,K4)=T
      DO 1200 I=1,NPLT
1200  DAT(I+1,K4)=Y(NPARM(I))
      CONTINUE
1201  CONTINUE
      if(DABS(t-(dt*k4+t0)).lt. .00001)k4=k4+1
      t=t+dt
      if(idf .eq. 0 .and. t .gt. tmax)go to 20
      if(idf .eq. 0)go to 10 .and. t .lt. tr)go to 10
      idf=0
      t=t0
      go to 139
20    continue
      IF(ID .NE. 1)GO TO 705
      DATA (NDIST(I),I=1,10)/10*0 0/
      DO 701 J=1,10) I=1,10)/100*0 0/
      DO 701 I=1,ND
      MDIST(I)=MDIST(I)+ADIST(I,J)*WD(I)
701   CONTINUE
      MDIST(I)=MDIST(I)/K4
702   CONTINUE
      DO 704 I=1,ND
      DO 704 J=1,ND
      DO 703 K=1,K4
      CDIST(I,J)=CDIST(I,J)+(ADIST(I,K)-MDIST(I))*(ADIST(J,K)-MDIST(J))
703   CONTINUE
      CDIST(I,J)=CDIST(I,J)/K4
704   CONTINUE
      WRITE(5,800)
800   FORMAT(/,2X,'  RANDOM DISTURBANCE MEAN VALUES ')
      WRITE(5,802)(MDIST(I),I=1,ND)
      WRITE(5,801)
801   FORMAT(/,2X,'  RANDOM DISTURBANCE COVARIANCE MATRIX ')
      DO 8000 I=1,ND
8000  WRITE(5,802)(CDIST(I,J),J=1,ND)
802   FORMAT(9(2X,1PD10.3))
      write(5,775)
775   format(/,2x,' SCALE FACTORS FOR RANDOM DISTURBANCES ')
      write(5,802)(sf(i),i=1,nd)
      data ((av(i,j),i=1,10),j=1,10)/100*0.0/
      do 902 i=1,no
      do 902 k=1,k4
      av(i,k)=av(i,k)/k4
902   continue
      do 903 k=1,k4
      do 903 i=1,no
      cav(i,j)=cav(i,j)+(av(i,k)-mav(i))*(av(j,k)-mav(j))
903   continue
      do 904 i=1,no
      do 904 j=1,no
      cav(i,j)=cav(i,j)/k4
      if(i.eq.j)rav(i)=sqrt(cav(i,i))
904   continue
      write(5,1000)
1000  format(/,2x,' OUTPUT MEAN VALUES ')
      write(5,802)(mav(i),i=1,no)
      write(5,1001)
1001  format(/,2x,' OUTPUT COVARIANCE MATRIX ')
      do 1002 i=1,no
1002  write(5,802)(cav(i,j),j=1,no)
      write(5,905)
905   format(/,2x,' RMS OUTPUT VALUES ')
      write(5,802)(rav(i),i=1,no)
C
C NOW CALL THE PLOTTING SUBROUTINE
C
705   NPNTS=k4-1
      call PLOTEM(DAT,NTYPE,NPLT,NPLTR,NPNTS,GRTIT,PLTIT)
      STOP
      end
C.....
C
C SUBROUTINE PLOTEM(DAT,NTYPE,NPLT,NPLTR,NPNTS,GRTIT,PLTIT)
C
C THIS SUBROUTINE LOTS UP TO 12 VARIABLES VERSUS TIME
C WITH EITHER 1,3,OR 4 PLOTS PER PAGE
C
C     DAT: AN ARRAY OF DIMENSION(13,5000) CONTAINING THE DATA TO
C          BE PLOTTED.  THE FIRST COLUMN CONTAINS THE TIME (X AXIS)
C          FOR ALL 12 OF THE DATA SETS(LOCATED IN COLUMNS 2 13 OF
C          DAT
C
C     NTYPE:   =1  FOR 1 PLOT PER PAGE
C              =3  FOR 3 PLOTS PER PAGE
C              =4  FOR 4 PLOTS PER PAGE
C
C     NPLT: THE TOTAL NUMBER OF PLOTS.  NPLT/NTYPE IS THE NUMBER
C          OF PAGES OF DATA AND MUST BE AN INTERGER
C
```

SIMPLOT Computer Program.

```
************************************************************************
******* PROGRAM DISCSM *************************************************
******* A DISCRETE TIME SIMULATION OF A LINEAR SYSTEM ******************
******* WRITTEN BY BRUCE E  GARDNER ************************************
******* LAST REVISED 11 FEB 83 BY RICK HOLDRIDGE    ********************
************************************************************************
        implicit real*8 (a h o z)

 NOTE   DAT IS ONLY SINGLE PRECISION

        REAL DAT(13.5000)
        INTEGER*4 L1(10)
        CHARACTER*50 CRTIT
        CHARACTER*20 PLTIT(12)
        dimension y(75).yp(75).tswi(20).delfi(20).opt(75)
        dimension dist(20).sf(20).wO(20)
        dimension fmat(40.40).gmat(40.6) hmat(20.40).dmat(20 6)
        dimension cfb(6.40)
        dimension wmat(40.20).dwmat(20.20).cdist(20.20).adist(20,6000)
        dimension u(6).NPARM(12)
        real mdist(20) av(20.6000).rav(20).mav(20).cav(20.20)

 READ THE PLOT DATA

        READ(30 *)NTYPE NPLT.NPLTR.DT1.TMAX.IDF.TR
        READ(30.*) (NPARM(I)).I=1.NPLT)
        READ(30 *)CRTIT (PLTIT(I).I=1 NPLT)
        read(01.*)ns.nc.nd.no dt
        read(01 *) ((fmat(i j)).j=1.ns).i=1.ns)
        if(no  eq  0)go to 1
        read(01.*) ((hmat(i j)).j=1 ns).i=1 no)
        read(01.*) ((dmat(i j)).j=1 nc).i=1.no)
        read(01.*) ((cfb(i.j)).j=1.ns).i=1.nc)
        if(nd  eq  0) go to 404
        read(01.*) ((wmat(i.j)).j=1.nd).i=1.ns)
        read(01 *) ((dwmat(i.j)).j=1.nd).i=1.no)
        read(01.*) (wO(i).i=1.nd)
        data  (dist(i).i=1.20)/20*0 0/
590     write(5.600)
600     format(/.2x.'IF      DIST  IS RANDOM. TYPE 1. IF NOT. TYPE 0 ')
        read(5.*.end=590)id
        IF(ID  EQ  0)GO TO 404
601     WRITE(5.602)
602     FORMAT(/.2X 'INPUT SCALE FACTORS FOR      DISTURBANCES ')
        READ(5.*.END=601)(SF(I) I=1 ND)
603     WRITE(5 604)
604     FORMAT(/.2X.'IF DESIRE NEW SEEDS. TYPE ANY ODD INTEGER. IF NOT.
     *TYPE O ')
        READ(5.*.END=603)ISEED

C    INITIALIZE SEEDS
        DO 4 I=1.10
        L1(I)=10000+2*I-1
        IF(ISEED EQ 0)GO TO 4
        L1(I)=L1(I)+ISEED
4       CONTINUE
C
C READ IN THE INITIAL CONDITIONS FOR THE SIMULATIONS
C
404     read(30.*)(y(i).i=1.ns)
21       do 86 i=1.nc+no
        y(ns+i)=0 0
86      continue
        k4 1
        t t0
        if(idf  ne  0)go to 78
101     format(1x f11 4 14(f12 4))
133     CONTINUE
78      do 6 i 1 nc
        do 6 k 1 ns
        x y(k)
        y(ns+i)=y(ns+i)+CFB(i.k)*x
6       continue
        do 79 i=1 nc
        u(i)=y(ns+i)
79      continue
        if(idf  ne  0)go to 10
        do 7 i 1.no
        do 7 k=1.ns
        y(ns+nc+i) y(ns+nc+i)+hmat(i.k)*y(k)
7       continue
        do 8 i=1.no
        do 8 k=1.nc
        y(ns+nc+i)=y(ns+nc+i)+dmat(i.k)*u(k)
8       continue
        if(nd eq 0)go to 9
        do 9 i=1 no
        do 9 k=1.nd
        y(ns+nc+i)=y(ns+nc+i)+dwmat(i.k)*wO(k)
9       continue
        do 200 i=1.no
        opt(i)=y(ns+nc+i)
200     continue
        do 201 i=1.nc
        opt(no+i)=u(i)
201     continue
C
C WRITE DATA TO THE DAT ARRAY
C
        DAT(1.K4)=T
        DO 1100 II=1.NPLT
                DAT(II+1.K4)=Y(NPARM(II))
 1100   CONTINUE
         t=t+dt
        K4=1
10      do 27 i=1.ns
        yp(i)=0 0
27      continue
        if(nd  le  0)go to 413
        if(id  eq  0)go to 22
        do 67 k=1.nd
        dist(k)=(RAN(L1(K))  5)/( 289102*SQRT(DT)*SF(K))
        if(idf  eq  0)adist(k.k4)=dist(k)
67      continue
22      do 30 i=1.ns
        do 30 k=1.nd
30      yp(i)=yp(i)+wmat(i.k)*(dist(k)+wO(k))
413     continue
        do 40 i=1.ns
        do 40 k=1.ns
40      yp(i)=yp(i)+fmat(i.k)*y(k)
70      do 71 i=1.ns
71      y(i)=yp(i)
        if(idf  ne  0)go to 133
        do 87 i=1.nc+no
        y(ns+i)=0 0
```

m- Number of controls.

p'' - Number of performance variables.

Item 3

GRTIT,PLTIT(NPLT)

where:

GRTIT- A 50 character variable containing the title which will be put on all *NPLT/NTYPE* pages of plots.

PLTIT- An array of *NPLT* 20 character variables containing the y-axis labels for the plots. The x-axis is always time.

Note: These character variables must be exactly 50 and 20 characters long.

Item 4

XC0$(n+r+ns)$

XC0 is an array containing $n+r+ns$ initial conditions on the simulation states in the following order:

State	Parameter
x_1	
$\downarrow$	plant states
x_n	
z_1	
$\downarrow$	compensator states
z_r	
w_1	
$\downarrow$	noise filter states
w_{ns}	

Description of "Plotdata" File inputs

Item 1

NTYPE,NPLT,NPLTR,DT1,TMAX,IDF,TR

where:

NTYPE- "1" for 1 plot per page, horizontal. "3" for 3 plots per page, vertical. "4" for 4 plots per page, horizontal.

NPLT- The total number of plots to be made *NPLT/NTYPE* must be an integer less than 13.

NPLTR- "1" for immediate results on a VT125 screen. "0" to create plot files, VECTR1.PLV and PARM.PLV.

DT1- The data point interval, *DT1/DT*(from RSANDY) must be an integer.

TMAX- Final time of the simulation, *TMAX/DT1* must be less than 5000.

IDF- "1" to run the simulation *TR* seconds prior to recording data for plotting. This is used to allow filtered noise to reach a steady covariance. "0" to start taking plot data at time= 0.

TR- The number of seconds to wait prior to recording and plotting data.

Item 2

Plot parameter ID's, the array *NPARM*

NPARM is an array containing *NPLT* identification numbers of the variables to be plotted. The parameters are identified in the following sequence:

ID number	Parameter
1	x_1
↓	↓ plant states
n	x_n
$n+1$	z_1
↓	↓ compensator states
$n+r$	z_r
$n+r+1$	w_1
↓	↓ noise filter states
$n+r+ns$	w_{ns}
$n+r+ns+1$	u_1
↓	↓ controls
$n+r+ns+m$	u_m
$n+r+ns+m+1$	y_1
↓	↓ performance variables(from RSANDY)
$n+r+ns+m+p''$	$y_{p''}$

where:

n- Order of the plant.

r- Order of the compensator.

ns- Number of states in all the noise filters.

Running SIMPLOT

@SIMPLOT Plotdata Simfile

where:

SIMPLOT- A VMS command file which runs the SIMPLOT.EXE file with *Plotdata* and *Simfile* as data files.

Plotdata- A users created file containing data used by SIMPLOT. The input item descriptions are shown on following pages.

Simfile- The simulation model file previously created by an RSANDY run.

The SIMPLOT program will ask 3 questions:

1. "Are the disturbances random?"

Type "1" for yes or "0" for no. If no noise was modeled in the RSANDY run which created the *Simfile*, you must type "0". If you had noise in the RSANDY run but want a clean time response, you can type "0" and no noise will show up in the time responses. If you answered "0" to this question, the simulation will be run. If you answered "1", the program will ask:

2. "Input new random number generator seed?"

Type in any odd integer for a new seed or type in zero for the default seed.

3. "Input scale factors for disturbances"

To force the simulation to be driven by noise having the same RMS as the RSANDY data, type "1,1,1...1" where there are as many 1's as there were random inputs in the RSANDY data, i.e. m'. If you want to scale the RMS of the noise, change the "1's" accordingly. For example, typing "2,.5" will make the first noise source twice the RMS of the RSANDY data and will make the second source half as large.

Appendix M.

SIMPLOT Computer Program

This appendix is the user's guide for the SIMPLOT computer program. The simulation part of the code was written by Bruce Gardner. It uses models created by the RSANDY program to run simulations of the closed loop designs. The program has the option of including the random disturbances used in the RSANDY design. Up to 12 variables can be plotted versus time and displayed on a VT125 screen or sent to a file for later plotting. The listing of the program and an example data set (Navion with full-order compensator from Chapter 2) are shown at the end of the appendix. The user's guide begins on the next page.

SCALEM2 Computer Program. (contd)

```
EXT I
2240 PRINT TAB(10),"THE N MATRIX IS"
2250 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8),FNR(N(I,J)),\ NEXT J\ PRINT\ N
EXT I
2260 PRINT
2270 PRINT TAB(10),"THE COMPUTER SCALED VARIABLE MATICES ARE"
2280 PRINT TAB(10),"THE COMPUTER SCALED B MATRIX IS"
2290 FOR I=1 TO NR\ FOR J=1 TO NM\ PRINT TAB(J*8),FNR(BCS(I,J)),\ NEXT J\ PRINT\
 NEXT I
2300 PRINT TAB(10),"THE COMPUTER SCALED C MATRIX IS"
2310 FOR I=1 TO NC\ FOR J=1 TO NR\ PRINT TAB(J*8),FNR(CCS(I,J)),\ NEXT J\ PRINT\
 NEXT I
2320 PRINT TAB(10),"THE COMPUTER SCALED N MATRIX IS"
2330 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8),FNR(NCS(I,J)),\ NEXT J\ PRINT\
 NEXT I
2340 PRINT
2350 PRINT TAB(10),"THE COMPLETELY SCALED MATRICES ARE"
2360 PRINT TAB(10),"THE AFINAL MATRIX IS"
2370 FOR I=1 TO NR\ PRINT TAB(I*8),FNR(AFINAL(I)),\ NEXT I
2380 PRINT TAB(10),"THE BFINAL MATRIX IS"
2390 FOR I=1 TO NR\ FOR J=1 TO NM\ PRINT TAB(J*8),FNR(BFINAL(I,J)),\ NEXT J\ PRI
NT\ NEXT I
2400 PRINT TAB(10),"THE CFINAL MATRIX IS"
2410 FOR I=1 TO NC\ FOR J=1 TO NR\ PRINT TAB(J*8),FNR(CFINAL(I,J)),\ NEXT J\ PRI
NT\ NEXT I
2420 PRINT TAB(10),"THE NFINAL MATRIX IS"
2430 FOR I=1 TO NC\ FOR J=1 TO NC\ PRINT TAB(J*8),FNR(NFINAL(I,J)),\ NEXT J\ PRI
NT\ NEXT I
2440 PRINT TAB(10),"THE IOIFINAL MATRIX IS"
2450 FOR I=1 TO NC\ PRINT TAB(I*8),FNR(IOIFINAL(I)),\ NEXT I
2460 PRINT TAB(10),"THE IOIIFINAL MATRIX IS"
2470 FOR I=1 TO NC\ PRINT TAB(I*8),FNR(IOIIFINAL(I)),\ NEXT I
2472 PRINT TAB(10),"THE IODFINAL MATRIX IS"
2474 FOR I=1 TO NC\ PRINT TAB(I*8),FNR(IODFINAL(I)),\ NEXT I
2480 PRINT
2490 PRINT "THE A MATRIX REQUIRES A LRTA'3 THEN STRAU IN 1819 ASSEMBLY"
2500 PRINT "THE B MATRIX REQUIRES A LRTA'";18-SCALEB2;"THEN STRAU"
2510 PRINT "THE C MATRIX REQUIRES A LRTA'";18-SCALEC2;"THEN STRAU"
2520 PRINT "THE N MATRIX REQUIRES A LRTA'";18-SCALEN2;"THEN STRAU"
2530 PRINT "THE IOI(1) ELEMENT REQUIRES A LRTA'";18-SCALEIOI(1),"THEN STRAU"
2540 PRINT "THE IOI(2) ELEMENT REQUIRES A LRTA'";18-SCALEIOI(2),"THEN STRAU"
2550 PRINT "THE IOI(3) ELEMENT REQUIRES A LRTA'";18-SCALEIOI(3),"THEN STRAU"
2560 PRINT "THE IOII(1) ELEMENT REQUIRES A LRTA'";18-SCALEIOII(1),"THEN STRAU"
2570 PRINT "THE IOII(2) ELEMENT REQUIRES A LRTA'";18-SCALEIOII(2),"THEN STRAU"
2580 PRINT "THE IOII(3) ELEMENT REQUIRES A LRTA'";18-SCALEIOII(3),"THEN STRAU"
2582 PRINT "THE IOD(1) ELEMENT REQUIRES A LRTA'";18-SCALEIOD(1),"THEN STRAU"
2584 PRINT "THE IOD(2) ELEMENT REQUIRES A LRTA'";18-SCALEIOD(2),"THEN STRAU"
2586 PRINT "THE IOD(3) ELEMENT REQUIRES A LRTA'";18-SCALEIOD(3),"THEN STRAU"
2590 END
```

SCALEM1 Computer Program Example Data.

```
"NR,NM,NC,TP"
6,4,2,.05
"A MATRIX(NR)"
-.23081,.89831,-.23642,1.17826,.43492,.88649
"B MATRIX(NR X NM) MEASUREMENTS DISTRIBUTION MATRIX "
-16.358,-456.87,-.2653,.0545&
-8.274,-230.9,-.15366,.01224&
10.224,195.99,.02073,-.00518&
2.39548,49.65,.00599,.00061&
2.749,160.16,.17429,.0093&
1.04027,-.01363,-.00002,.00428
"C MATRIX(NC X NR) COMPENSATOR STATE TO CONTROL DISTRIBUTION MATRIX "
0.0,1.0,0.0,1.0,1.0, 68818&
-.91396,2.70164,-.90986,4.64211, 56013,1.0
"N MATRIX(NC X NC) COMMAND DISTRIBUTION MATRIX"
-.08369,-.03789&
-.164,.07336
"KI MATRIX(NC X NC) INTEGRAL GAIN MATRIX"
-.0042511,-.04904&
-.0040024,-.042269
"BSCALE(NM)------- THE SCALING FACTORS ON SPECIFIC COLUMNS OF B."
.017452,.017452,1,1
"CSCALE(NR)------- THE SCALING FACTORS ON SPECIFIC COLUMNS OF C."
1,1,1,1,1,1
"NSCALE(NC)------- THE SCALING FACTORS ON SPECIFIC COLUMNS OF N."
1,1
"KISCALE(NC)------ THE SCALING FACTORS ON SPECIFIC COLUMNS OF KI "
1,1
"YSCALE(NM)------- THE SCALING ON EACH MEASUREMENT."
500,500,32,8
"CNTRLSCALE(NC)--- THE COMPUTER SCALING ON EACH CONTROL."
1024,1024
"YDSCALE(NC)------ THE COMPUTER SCALING ON EACH DESIRED OUTPUT."
8,32
"YMAX(NM)--------- THE VECTOR OF MAX VALUES OF THE MEASUREMENTS."
20,20,20,30
"YDMAX(NC)-------- THE VECTOR OF MAX VALUES OF THE DESIRED OUTPUTS."
30,20
```

SCALEM2 Computer Program Example Data.

```
"NR,NM,NC,TP"
5,8,3,.05
"THE MINIMAL DISCRETE COMPENSATOR DYNAMICS MATRIX, PHI, IS:"
-0.78263,1.74302,0.90592,0.96006,0.72946
"THE MINIMAL DISCRETE COMPENSATOR INPUT MATRIX, GAMMAMIN, IS:"
-0.00068,0.00024,-0.00074,-0.01160,0.04977,0.01823,0.10458,-1.91023&
-0.00076,-0.00083,-0.00088,-0.01252,0.04972,0.00800,0.08074,-1.75084&
0.00214,-0.00052,0.00084,-0.00071,-0.10784,-0.05930,-0.00479,0.37056&
-0.00048,0.00001,0.00156,0.00007,0.00701,0.00129,0.00276,0.00769&
-0.00356,0.00115,0.00006,0.00581,-0.67143,0.13067,-2.77471,-1.64643
"THE MINIMAL COMPENSATOR OUTPUT MATRIX, HMIN, IS:"
-1.79786,1.82597,1.00000,-0.16434,1.00000&
0.10394,-0.03719,0.36894,1.00000,0.11184&
0.00000,1.00000,0.79317,0.03578,0.09061
"N MATRIX(NC X NC) COMMAND DISTRIBUTION MATRIX"
-4.829E-02,-9.785E-03,-2.378E-03&
7.004E-03,-7.883E-02,2.098E-03&
-1.175E-02,-3.420E-03,5.886E-02
"THE KX MATRIX"
-.26522,-.19939,-1.2212
"THE KXI MATRIX"
-.0000864,-.0001028,-.17365
"THE KXD MATRIX"
-.094118,-.72424,-3.0
"BSCALE(NM)------- THE SCALING FACTORS ON SPECIFIC COLUMNS OF B."
1.,1.,1.,.017452,.017452,.017452,.017452,.017452
"CSCALE(NR)------- THE SCALING FACTORS ON SPECIFIC COLUMNS OF C."
1.,1.,1.,1.
"NSCALE(NC)------- THE SCALING FACTORS ON SPECIFIC COLUMNS OF N."
1.,1.,1.
"YSCALE(NM)------- THE SCALING ON EACH MEASUREMENT."
32.,32.,32.,500.,500.,500.,500.,500
"CNTRLSCALE(NC)--- THE COMPUTER SCALING ON EACH CONTROL."
1024.,1024.,1024.
"YDSCALE(NC)------ THE COMPUTER SCALING ON EACH DESIRED OUTPUT."
32.,32.,32.
"YMAX(NM)--------- THE VECTOR OF MAX VALUES OF THE MEASUREMENTS."
50.,50.,30.,10.,10.,30.,20.,30
"YDMAX(NC)-------- THE VECTOR OF MAX VALUES OF THE DESIRED OUTPUTS."
20.,20.,20
```

END

FILMED

5-85

DTIC